1. _ChangeTracking . . . 3
2. _IncludeLibrary . . . 3
2.1 _FacebookTwitterGoogle_ButtonsCode_ . . . 4
2.2 _LearnCocos2DBookThumbnail . . . 4
2.3 _NotYetReleasedDisclaimer . . . 4
2.4 _scrap migrating cocos2d project . . . 4
2.5 _Template-Images . . . 5
2.5.1 _Cocos2D-With-UIKit-Views-Template_Images . . . 5
2.5.2 _Doodle-Drop-Template_Images . . . 6
2.5.3 _Hello-Cocos3D-Template_Images . . . 8
2.5.4 _Hello-Kobold2D-Template_Images . . . 10
2.5.5 _IncomingNetworkConnections . . . 12
2.5.6 _Isometric-Tilemap-Template_Images . . . 12
2.5.7 _Orthogonal-Tilemap-Template_Images . . . 13
2.5.8 _Parallax-Side-Scroller-Template_Images . . . 14
2.5.9 _Particle-Effects-Template_Images . . . 15
2.5.10 _Physics-Template_Images . . . 16
2.5.11 _Pinball-Game-Template_Images . . . 17
2.5.12 _Sprite-Performance-Template_Images . . . 19
2.5.13 _User-Input-Template_Images . . . 20
2.6 _TemplateProjects . . . 21
2.6.1 _AdvancedTemplateProjects . . . 21
2.6.2 _BasicTemplateProjects . . . 24
2.6.3 _GameTemplateProjects . . . 26
3. Kobold2D Documentation . . . 28
3.1 Kobold2D User's Guide . . . 29
3.1.1 Kobold2D Requirements . . . 29
3.1.2 Installing Kobold2D . . . 29
3.1.3 Creating a Kobold2D Project . . . 30
3.1.4 Working with Kobold2D Projects . . . 32
3.1.5 Upgrading Kobold2D Projects . . . 33
3.1.6 Migrating a Cocos2D Project . . . 35
3.1.7 The Kobold2D Template Projects . . . 44
3.1.8 Creating a Kobold2D Template Project . . . 51
3.1.9 Installing a Kobold2D Template Project . . . 53
3.1.10 Xcode Build Locations Setup for Kobold2D . . . 53
3.2 Kobold2D Programming Guide . . . 56
3.2.1 Config.lua Settings Reference . . . 56
3.2.2 Working With Ad Banners . . . 57
3.2.3 Processing User Input . . . 59
3.2.3.1 KKInput General Information . . . 59
3.2.3.2 KKInput Touches . . . 60
3.2.3.3 KKInput Gestures . . . 62
3.2.3.4 KKInput Accelerometer . . . 66
3.2.3.5 KKInput Gyroscope . . . 67
3.2.3.6 KKInput Device Motion . . . 68
3.2.3.7 KKInput Keyboard . . . 69
3.2.3.8 KKInput Mouse . . . 72
3.2.4 Working with Automatic Reference Counting (ARC) . . . 73
3.2.4.1 ARC Requirements for Development and Deployment . . . 74
3.2.4.2 What you need to know about ARC . . . 75
3.2.4.3 ARC with C or C++ code (Chipmunk, Box2D) . . . 76
3.2.4.4 Enabling ARC in legacy Kobold2D projects . . . 78
3.2.4.5 Disabling Automatic Reference Counting . . . 79
3.3 Kobold2D Reference Documentation . . . 80
3.3.1 API References . . . 80
3.3.1.1 Box2D API Reference . . . 80
3.3.1.2 Chipmunk API Reference . . . 80
3.3.1.3 Chipmunk SpaceManager API Reference . . . 80
3.3.1.4 Cocos2D API Reference (iOS) . . . 80
3.3.1.5 Cocos2D API Reference (Mac OS) . . . 81
3.3.1.6 Cocos2D Extensions API Reference (iOS) . . . 81
3.3.1.7 Cocos2D Extensions API Reference (Mac OS) . . . 81
3.3.1.8 CocosDenshion API Reference (iOS) . . . 81
3.3.1.9 CocosDenshion API Reference (Mac OS) . . . 81
3.3.1.10 Kobold2D API Reference (iOS) . . . 81
3.3.1.11 Kobold2D API Reference (Mac OS) . . . 81
3.3.1.12 ObjectAL API Reference (iOS) . . . 81
3.3.1.13 SneakyInput API Reference . . . 81
3.3.2 Cocos2D Resources . . . 81
3.3.3 Lua Reference Manual . . . 82
3.4 Kobold2D Contributor's Guide . . . 82
3.4.1 Kobold2D Build Status . . . 83
3.5 Kobold2D FAQ . . . 83
3.5.1 Kobold2D General FAQ . . . 84

3.5.1.1 Why was Kobold2D created? . . . 85
3.5.1.2 Why isn't Kobold2D a contribution to cocos2d-iphone? . . . 86
3.5.1.3 Should I try Cocos2D first before switching to Kobold2D? . . . 86
3.5.1.4 Do I need lots of Cocos2D experience to start with Kobold2D? . . . 86
3.5.1.5 Do all the Cocos2D tutorials and books work with Kobold2D? . . . 86
3.5.1.6 When was Kobold2D created? . . . 87
3.5.1.7 What does Kobold2D cost? . . . 87
3.5.1.8 Can I donate to Kobold2D? . . . 87
3.5.1.9 Why does Kobold2D consume 400 MB disk space? . . . 87
3.5.1.10 How can I reduce the installed size of Kobold2D? . . . 87
3.5.1.11 How can I install Kobold2D to a custom folder? . . . 88
3.5.1.12 How can I uninstall Kobold2D? . . . 88
3.5.2 Kobold2D Xcode FAQ . . . 88
3.5.2.1 Why does Kobold2D build successfully but won't run? . . . 88
3.5.2.2 Why do I get "ld .. failed with exit code 1"? . . . 92
3.5.2.3 Is Kobold2D compatible with Xcode 3? . . . 92
3.5.2.4 Does Kobold2D work with the latest beta SDK? . . . 92
3.5.2.5 How can I create a blank Kobold2D project? . . . 93
3.5.2.6 Can I add the Kobold2D.xcodeproj to my project? . . . 93
3.5.2.7 Can I work in a copy of the Kobold2D Workspace? . . . 93
3.5.2.8 Why does Xcode crash after installing Kobold2D or the Xcode Help files? . . . 94
3.5.2.9 How can I copy the Build Log in Xcode? . . . 94
3.5.3 Kobold2D Technical FAQ . . . 94
3.5.3.1 Can Kobold2D Projects be located in a custom folder? . . . 94
3.5.3.2 Where does the 'Network Connections' dialog come from? . . . 95
3.5.3.3 How to disable iSimulate? . . . 96
3.5.3.4 Why is the AppDelegate class empty? . . . 96
3.5.3.5 Are Kobold2D apps larger than pure cocos2d-iphone apps? . . . 96
3.5.3.6 How to fix compile errors with CocosBuilder? . . . 97
3.5.3.7 Why does Kobold2D compile libraries I don't use? . . . 98
3.5.4 Kobold2D Cocos2D FAQ . . . 98
3.5.4.1 Is Kobold2D compatible with cocos2d-iphone? . . . 98
3.5.4.2 Is Kobold2D updated when cocos2d-iphone is? . . . 98
3.5.4.3 Will installing Kobold2D affect my Cocos2D Projects? . . . 98
3.5.5 Kobold2D Wax & Lua FAQ . . . 98
3.5.5.1 What's the difference between Wax and Corona SDK? . . . 98
3.5.5.2 Can I write game logic in Lua with Kobold2D? . . . 100
3.5.6 Kobold2D Audio FAQ . . . 100
3.5.6.1 Which audio engine is better - CocosDenshion or ObjectAL? . . . 100
3.5.7 Kobold2D Physics FAQ . . . 100
3.5.7.1 Which Physics Engine is the best? . . . 100
3.5.8 Kobold2D Cocos3D FAQ . . . 101
3.5.8.1 Is there a Mac OS version of Cocos3D? . . . 101
3.6 Kobold2D Release Notes . . . 102
3.6.1 Kobold2D v0.9.1 Release Notes . . . 102
3.6.2 Kobold2D v0.9.2 Release Notes . . . 103
3.6.3 Kobold2D v0.9.3 Release Notes . . . 103
3.6.4 Kobold2D v0.9.4 Release Notes . . . 103
3.6.5 Kobold2D v0.9.5 Release Notes . . . 104
3.6.6 Kobold2D v0.9.6 Release Notes . . . 104
3.6.7 Kobold2D v1.0.0 Release Notes . . . 105
3.6.8 Kobold2D v1.0.1 Release Notes . . . 105
3.6.9 Kobold2D v1.0.2 Release Notes . . . 106
3.6.10 Kobold2D v1.0.4 Release Notes . . . 107
3.6.11 Kobold2D v1.0.5 Release Notes . . . 107
3.6.12 Kobold2D v1.1.0 Release Notes . . . 108
3.6.13 Kobold2D v1.1.1 Release Notes . . . 108
3.6.14 Kobold2D v1.1.2 Release Notes . . . 109
3.6.15 Kobold2D v1.1.3 Release Notes . . . 109
3.6.16 Kobold2D v1.x.x Release Notes . . . 109
3.6.17 Kobold2D v2.0.0 Release Notes . . . 110
3.6.18 Kobold2D v2.0.1 Release Notes . . . 110
3.6.19 Kobold2D v2.0.2 Release Notes . . . 111
3.6.20 Kobold2D v2.0.3 Release Notes . . . 111
3.6.21 Kobold2D v2.0.4 Release Notes . . . 111
3.6.22 Kobold2D v2.1.0 Release Notes . . . 111
3.6.23 Kobold2D v2.x.x Release Notes . . . 112

_ChangeTracking

Recently Updated

- Which Physics Engine is the best?
  updated by Steffen Itterheim
  (view change)
  May 12, 2013
- Enabling ARC in legacy Kobold2D projects
  updated by Steffen Itterheim
  (view change)
  Mar 21, 2013
- Kobold2D Release Notes
  updated by Steffen Itterheim
  (view change)
  Feb 23, 2013
- Kobold2D v2.1.0 Release Notes
  updated by Steffen Itterheim
  (view change)
  Feb 23, 2013
- Kobold2D v1.1.3 Release Notes
  updated by Steffen Itterheim
  (view change)
  Sep 22, 2012
- Kobold2D v2.0.4 Release Notes
  updated by Steffen Itterheim
  (view change)
  Sep 22, 2012
- Kobold2D v1.1.2 Release Notes
  created by Steffen Itterheim
  Jul 27, 2012
- Kobold2D v2.0.3 Release Notes
  updated by Steffen Itterheim
  (view change)
  Jul 27, 2012
- _FacebookTwitterGoogle_ButtonsCode_
  created by Steffen Itterheim
  Jul 15, 2012
- Kobold2D v2.0.2 Release Notes
  created by Steffen Itterheim
  Jul 12, 2012
- Kobold2D v2.0.1 Release Notes
  updated by Steffen Itterheim
  (view change)
  Jul 07, 2012
- Kobold2D v1.1.1 Release Notes
  updated by Steffen Itterheim
  (view change)
  Jul 07, 2012
- Kobold2D v2.x.x Release Notes
  updated by Steffen Itterheim
  (view change)
  Jul 07, 2012
- Kobold2D v1.x.x Release Notes
  updated by Steffen Itterheim
  (view change)
  Jul 07, 2012
- Kobold2D v1.0.0 Release Notes
  updated by Steffen Itterheim
  (view change)
  Jul 05, 2012
- More

_IncludeLibrary

_FacebookTwitterGoogle_ButtonsCode_

// header

Tweet

// footer

Tweet

_LearnCocos2DBookThumbnail

_NotYetReleasedDisclaimer

This Kobold2D version is not available yet!
This page describes changes to Kobold2D which have been implemented, but the Kobold2D version it refers to has not been released yet.

_scrap migrating cocos2d project

Make a backup of your project just in case!

This is a general guide showing you the principles of converting an existing Cocos2D project to Kobold2D. It does not offer a solution to every possibly required change to successfully migrate a project. Each project is different and may require custom modifications not outlined in this guide.

1. create a kobold2d project from Hello Kobold2D template project
    - remove unneeded files:
        a. remove HelloWorldScene class (.h/.m)
        b. remove platform target (iOS or Mac) that isn't supported by your project
        c. remove platform specific scheme (iOS or Mac) that isn't supported by your project
        d. remove the platform-specific Projectfiles-XXX group that isn't supported by your project
        e. remove unneeded Resource files:
            - ship.png
            - ship-hd.png
            - Pow.caf
2. modify config.lua:
    - change FirstSceneClassName to your project first scene's name

        In that scene the class method "+(id) scene" is not called. Any required code in that class needs to be moved to the scene's init method.

    - change any other settings of your project deviating from the default settings, for example Orientation, Color format, Pixel format, Retina support, etc.

> ⚠ This is where most issues will arise when migrating. Be sure to check all the config.lua settings for changes regarding cocos2d startup.

3. Add Files to "project name"
    a. select all source code files, except:
        - AppDelegate
        - RootViewController
        - GameConfig.h
        - any other library source code that is already provided by Kobold2D (eg. Cocos2D, etc.)
    b. make sure to check "copy to destination"
4. Add Files to "project name"
    a. select all of your project's unique resources, be sure not to include:
        - Info.plist
        - fps_images.png
        - any Cocos2D resources that you did not modify, for example: Icon.png, Default.png, ...
    b. make sure to check "copy to destination"

_Template-Images

- _Cocos2D-With-UIKit-Views-Template_Images
- _Doodle-Drop-Template_Images
- _Hello-Cocos3D-Template_Images
- _Hello-Kobold2D-Template_Images
- _IncomingNetworkConnections
- _Isometric-Tilemap-Template_Images
- _Orthogonal-Tilemap-Template_Images
- _Parallax-Side-Scroller-Template_Images
- _Particle-Effects-Template_Images
- _Physics-Template_Images
- _Pinball-Game-Template_Images
- _Sprite-Performance-Template_Images
- _User-Input-Template_Images

_Cocos2D-With-UIKit-Views-Template_Images

_Doodle-Drop-Template_Images

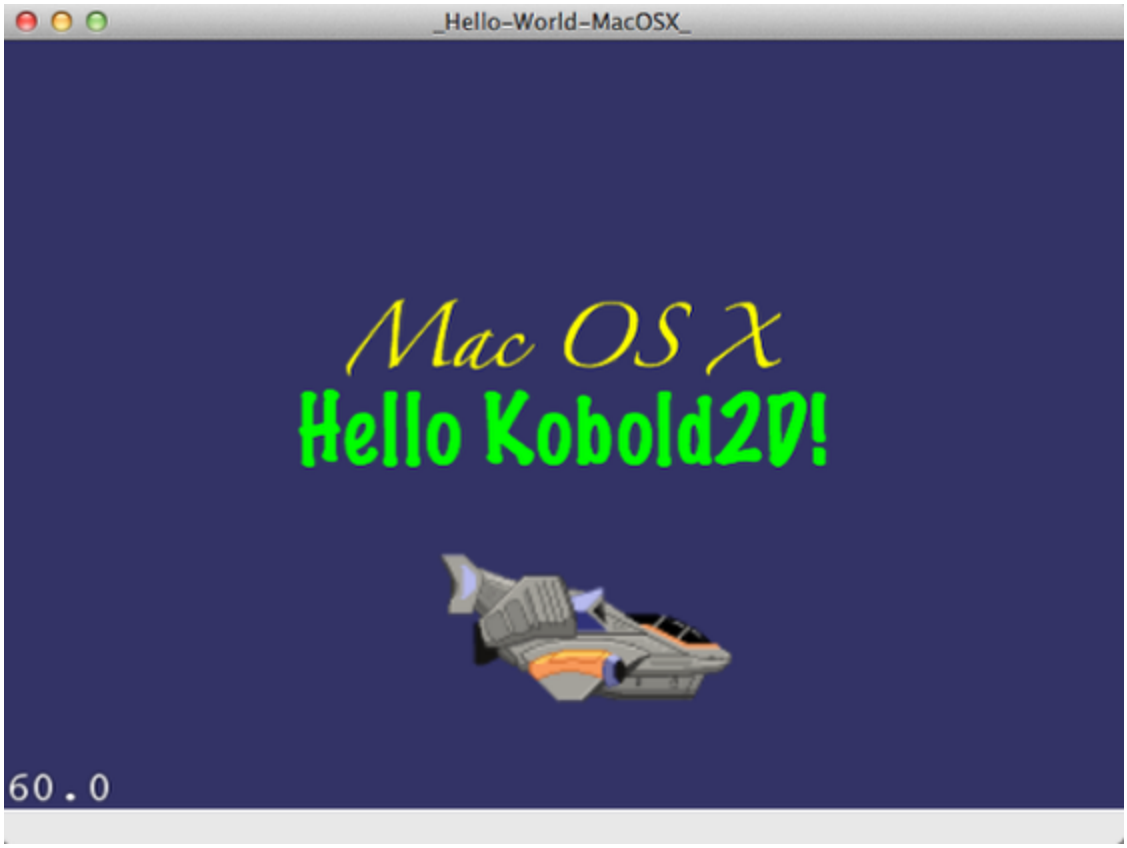
11
GAME OVER!

_Hello-Cocos3D-Template_Images

Hello Kobold2D!
60.0

_Hello-Kobold2D-Template_Images

Test Advertisement
iAd
iPhone/iPod Touch Simulator
Hello Kobold2D!
59.9



_IncomingNetworkConnections

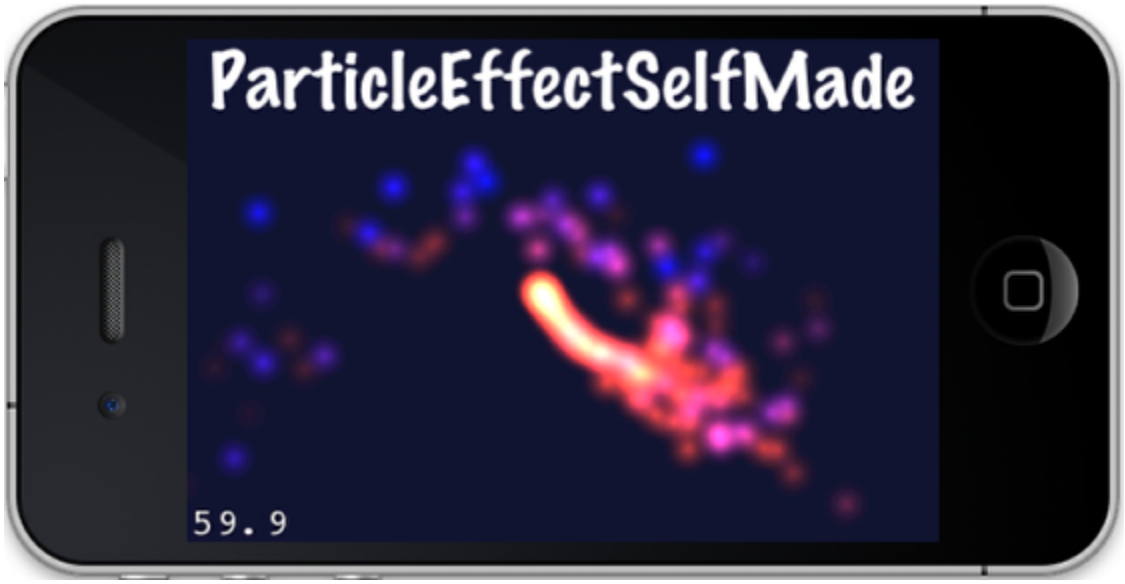
Do you want the application "_Doodle-Drop-Template_-iOS" to accept incoming network connections?
Clicking Deny may limit the application's behavior. This setting can be changed in the Firewall pane of Security preferences.
Deny
Allow

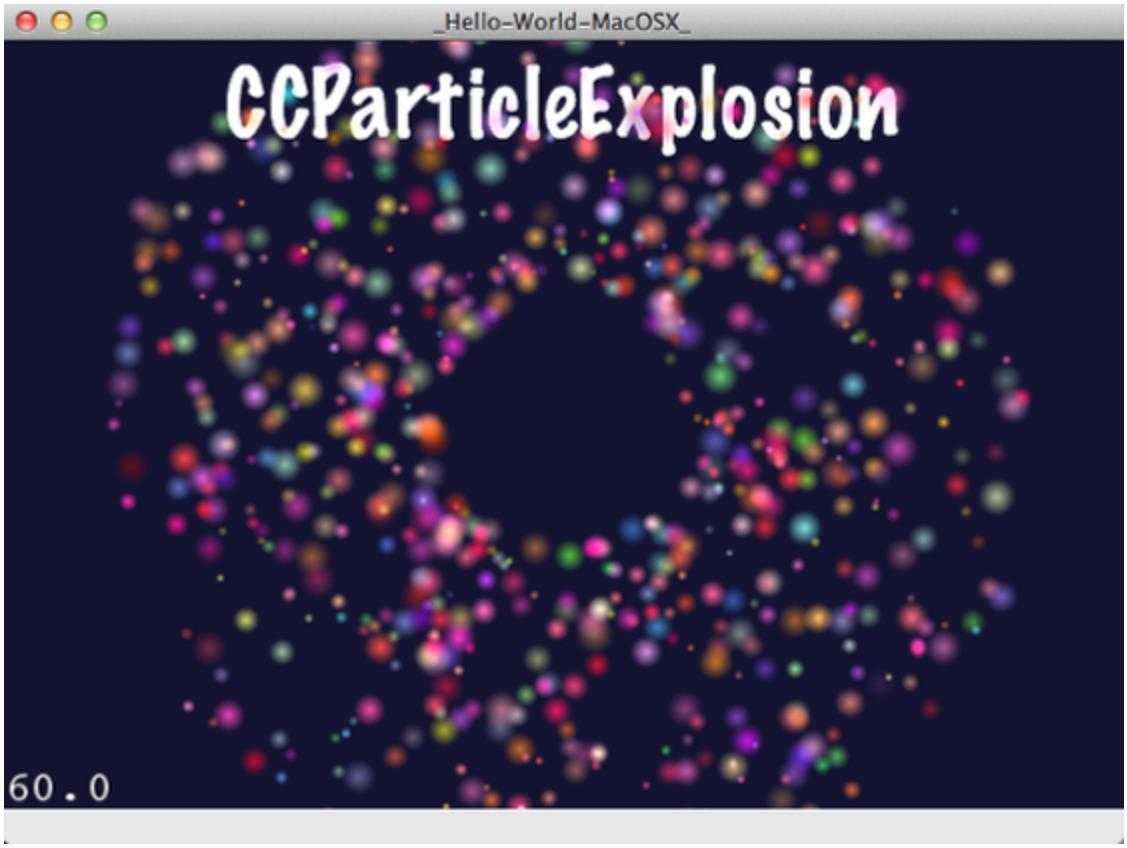
_Isometric-Tilemap-Template_Images

_Orthogonal-Tilemap-Template_Images

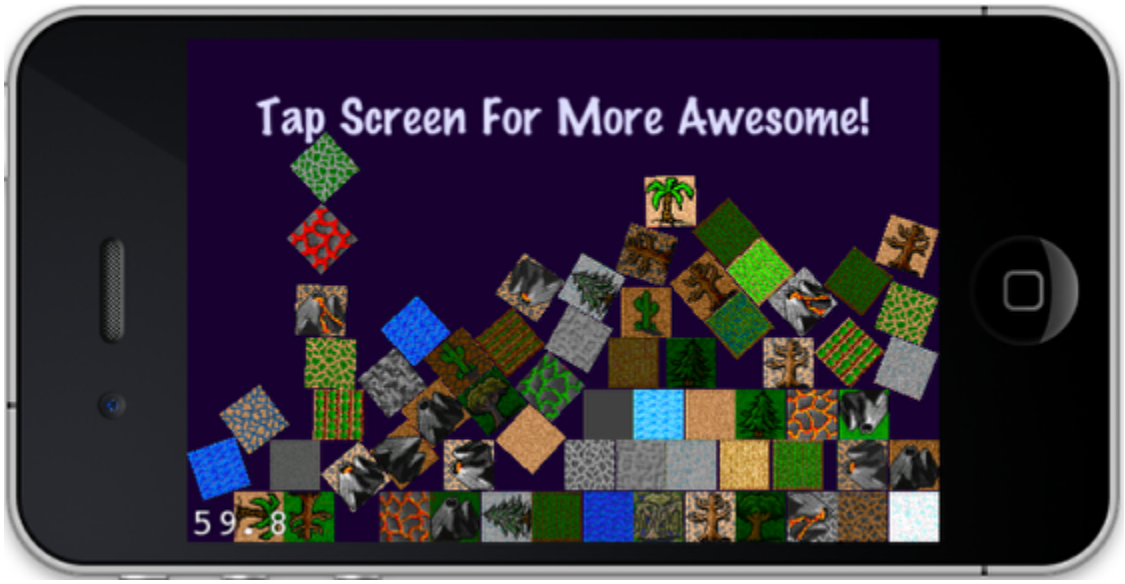
_Parallax-Side-Scroller-Template_Images

_Particle-Effects-Template_Images

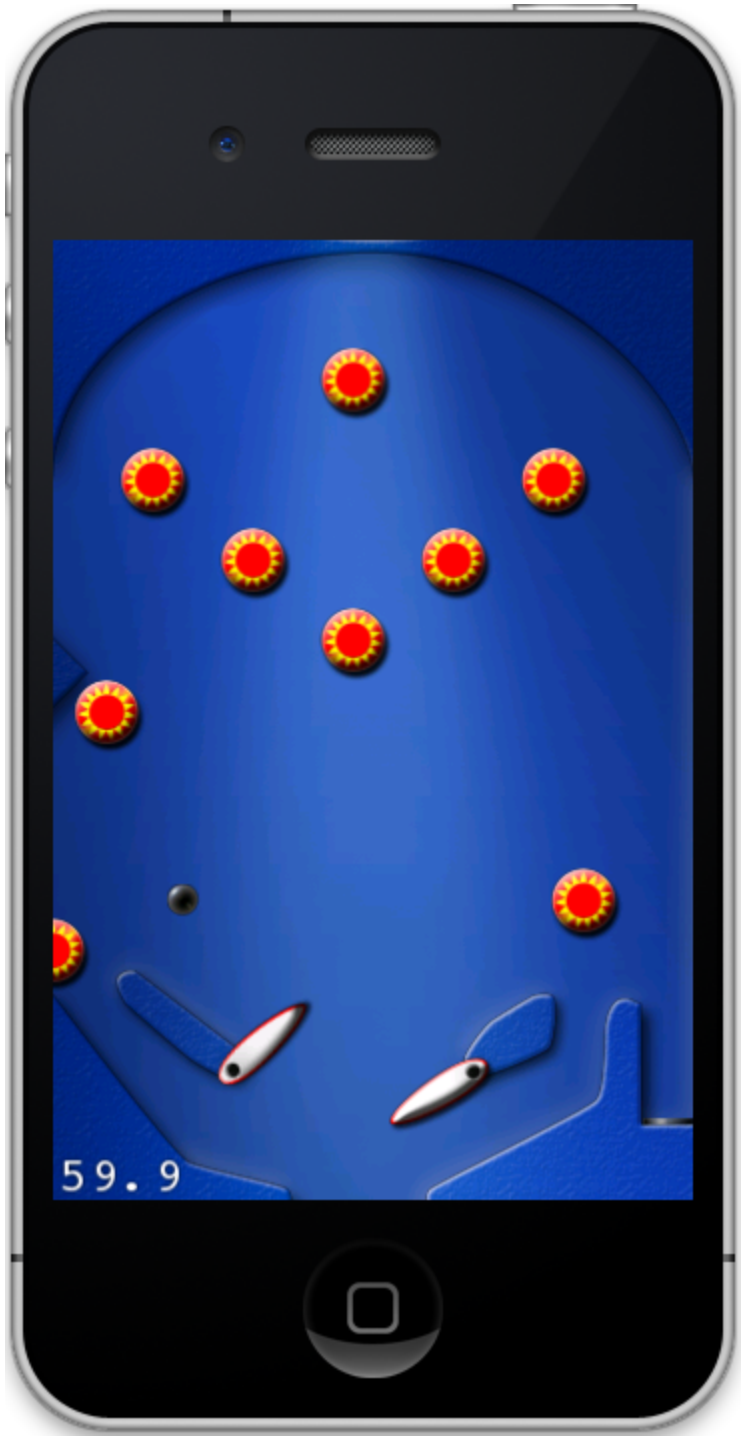


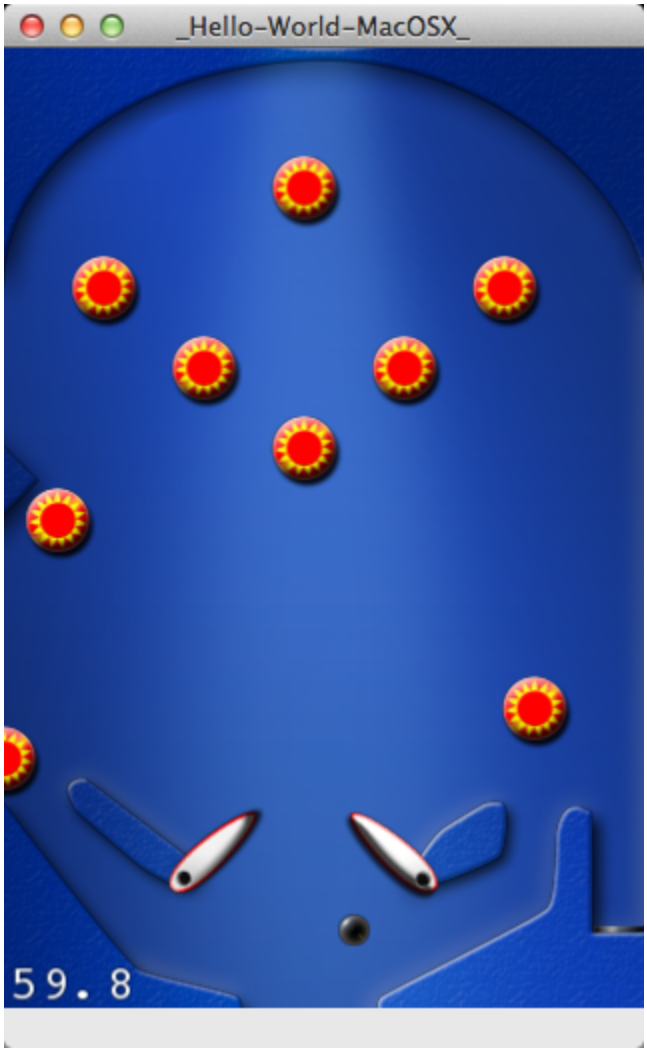


_Physics-Template_Images

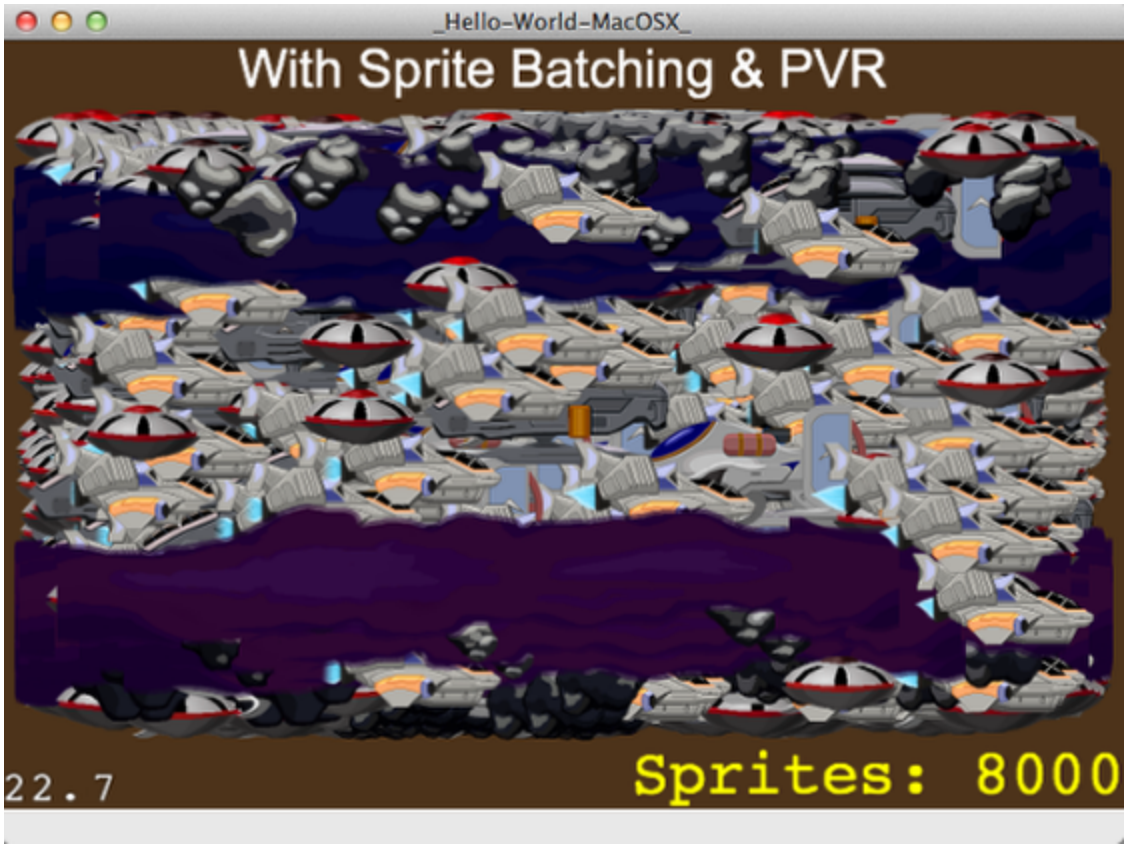


_Pinball-Game-Template_Images

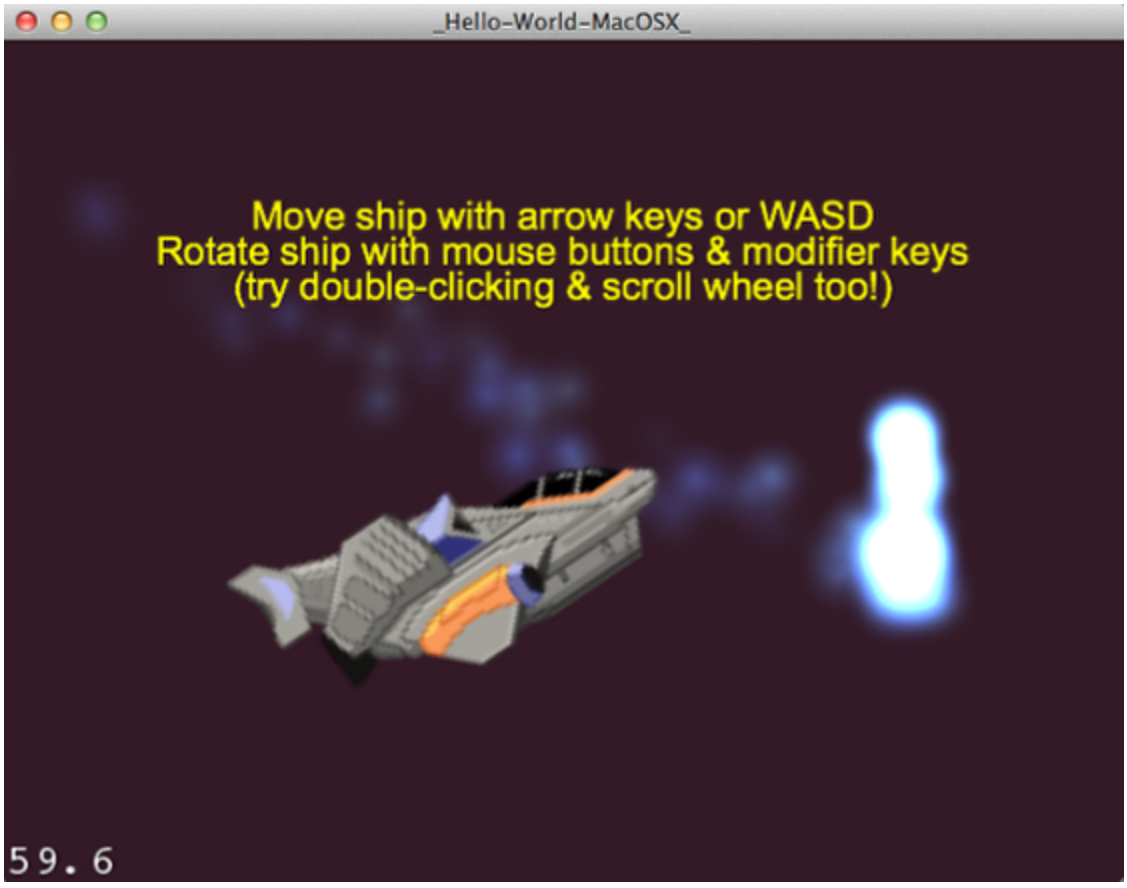

59.9

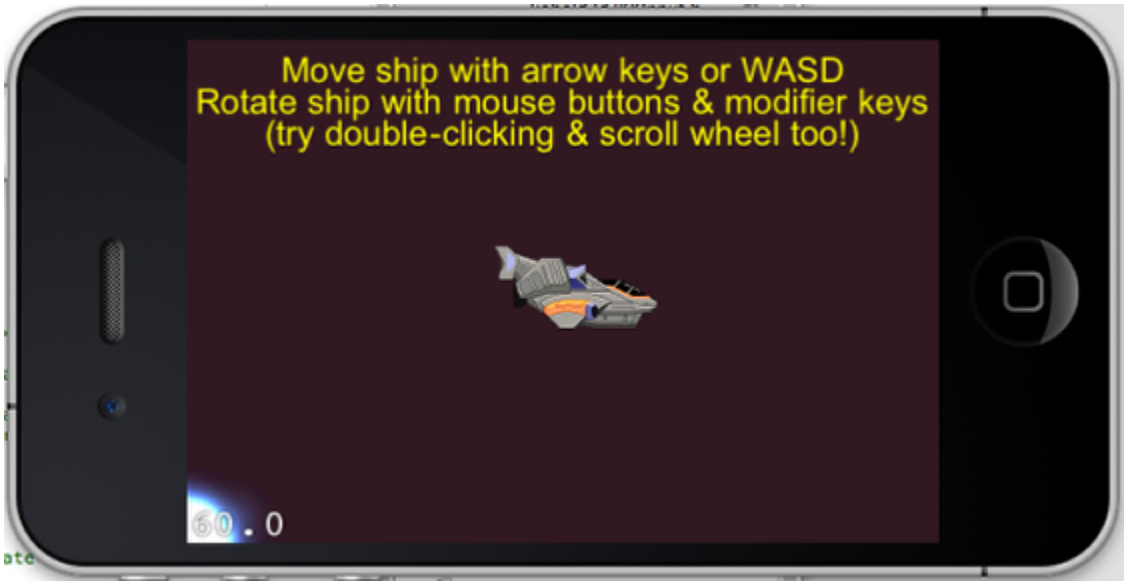


_Sprite-Performance-Template_Images



_User-Input-Template_Images


Hello-World-MacOSX
Move ship with arrow keys or WASD
Rotate ship with mouse buttons & modifier keys
(try double-clicking & scroll wheel too!)
59.6



_TemplateProjects

- _AdvancedTemplateProjects
- _BasicTemplateProjects
- _GameTemplateProjects

_AdvancedTemplateProjects

Physics Box2D / Physics Chipmunk / Physics Chipmunk SpaceManager

iOS

Mac OS

The three physics template projects in Kobold2D all implement the same physics demo: adding dynamic bodies (boxes), collision callbacks, and the screen/window acting as an impenetrable barrier. A static body is used to attach three dynamic bodies which are connected via Revolute (Box2D) / Pivot (Chipmunk) joints to create a swinging "rope".

Chapter 12

If you're unsure which physics engine is right for you, check out these templates and use the one which you find easiest to work with. You might also want to read the FAQ entry: Which Physics Engine is the best?

Orthogonal Tilemap

iOS

Mac OS

The orthogonal tilemap project shows how to load a TMX tilemap created with the Tiled Map Editor. It demonstrates conversion of touches to tile coordinates, inspecting tile properties, the use of the Tiled Object Layer for trigger areas and scrolling of the tilemap.

Chapter 10

Isometric Tilemap

iOS

Mac OS

The isometric tilemap project shows how to load an isometric TMX tilemap created with the Tiled Map Editor. The isometric tilemap is drawn with depth buffering and 2D projection for correct Z ordering of tiles. The project demonstrates conversion of touches to tile coordinates, inspecting tile properties, the use of the Tiled Object Layer for trigger areas and scrolling of the tilemap. It includes a controllable player character. On Mac OS it illustrates how to process keyboard events.

Chapter 11

Sprite Performance

iOS

Mac OS

This project illustrates how to improve rendering performance by replacing sprites created from individual files to sprites created from sprite frame provided by a PVR compressed texture atlas. It also shows how to use CCSpriteBatchNode to improve performance even further.

Blog Post

This project can also be used for benchmarking sprite rendering performance of various sizes and image formats.

Cocos2D With UIKit Views

iOS

Mac version currently has unresolved issues. See comments in AppDelegate.m.

Mac OS

This project shows you how to add Cocoa views to Cocos2D. In particular the project not only adds views on top of Cocos2D but also underneath (transparent OpenGL view). It also correctly performs hit testing on views to emulate the responder chain for Cocos2D nodes. This is achieved by enabling `EnableGLViewNodeHitTesting` in config.lua. The user can interact with views in the foreground, cocos2d nodes in the OpenGL view, and views behind the cocos2d OpenGL view.

Chapter 15

The Learn Cocos2D (2nd Edition) book Chapter 15 includes instructions on how to add Cocos2D to an existing UIKit app.

_BasicTemplateProjects

Hello Kobold2D

iOS

Mac OS

The standard template project for Kobold2D is the successor of the "Hello Cocos2D" template. It showcases Retina support, config.lua settings, OS and device detection, iAd banners, actions, built-in and custom font labels and sound.

Chapter 16

Use this template as a starting point if you want to start a Kobold2D project from scratch.

Hello Cocos3D

iOS

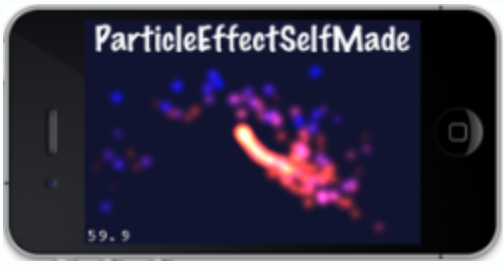
iOS

This template is similar to *Hello Kobold2D* and showcases the use of Cocos3D, mixed with regular Cocos2D nodes. Notice that Cocos2D nodes can be drawn in front of or behind 3D objects. Cocos3D currently does not support Mac builds.

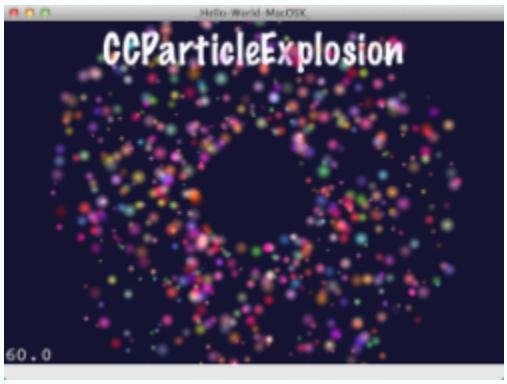
Chapter 16

Use this template as a starting point if you plan to work with Cocos3D. It has the required line `-force_load libcocos3d.a` in `BuildSettings-iOS.xcconfig` already set.

Particle Effects



iOS



Mac OS

This project shows you how to create particle effects by subclassing the CCParticleSystem class. It also displayed particle effects created with Particle Designer as well as Cocos2D's built-in particle effects.

Chapter 9

User Input

iOS

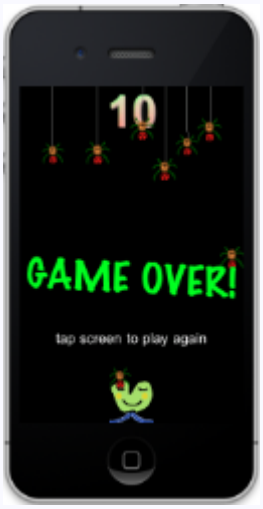
Mac OS

Illustrates how to use the KKInput class which wraps user input into a collection of convenience method. The main difference being that KKInput allows you to poll the input state at any time from any class without having to register to receive input events. KKInput takes the grunt work out of input processing, giving you the ability to easily test for double-clicks, scroll wheel & mouse position changes, key states with or without modifiers, and more.

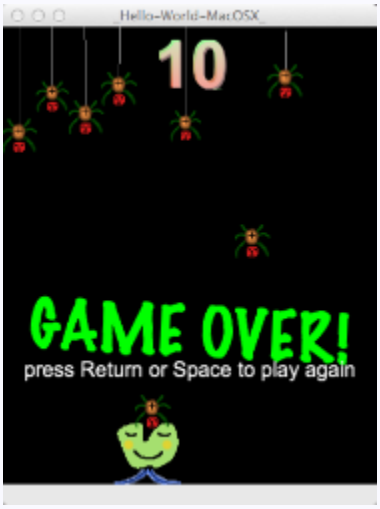
Exclusive Kobold2D Feature!

_GameTemplateProjects

Doodle Drop



iOS



Mac OS

Doodle Drop is a fully playable game in which you attempt to avoid the dropping spiders by tilting your device. The game showcases accelerometer input (keyboard input on Mac), sound, music, a game over screen, a custom loading screen, score counting wth a bitmap font label, object pooling to avoid frequent alloc/dealloc cycles and advanced use of easing, rendering lines, simple collision testing, sequence and call func actions.

Chapter 4

Parallax Side-Scroller

iOS

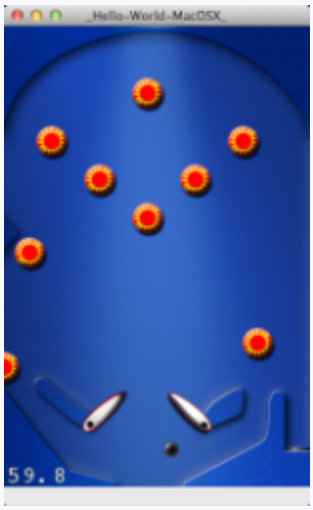
(no Mac OS version)

The Parallax Side-Scroller is a classic shoot'em up game with a virtual joypad and button provided by SneakyInput. The project shows you how to create an endlessly scrolling parallax background, how to structure bigger projects into multiple classes, how to pool and re-use game objects, how to keep nodes within a defined area by overriding setPosition, how to create reusable game components, how to turn your game's scene into a Singleton to make it accessible to other classes, and how to access other classes through your game's scene.

Chapters 6-8

Pinball Game

iOS



Mac OS

The Pinball project is a functional pinball table with bumpers, flippers, a plunger and a ball. Box2D is used for physics simulation, the collision shapes were created with PhysicsEditor. The plunger and flippers are animated via Box2D motors. Collision events are forwarded via Objective-C messages to individual objects and correspondingly named selectors (eg beginContactWithBall, endContactWithPlunger). Includes a demonstration of gravitational pull by uncommenting the function applyForceTowardsFinger.

Chapter 13

Kobold2D Documentation

User's Guide

The Kobold2D User's Guide explains how to work with Kobold2D: installation, creating and updating projects, migrating from Cocos2D, creating and installing Kobold2D template projects and the description for each pre-installed template project.

Programming Guide

The Kobold2D Programming Guide contains a description of all code features ofKobold2D. All the improvements over Cocos2D are described as well as the Kobold2D exclusive features, such as user input processing, banner ads and Lua support.

Reference Documentation

The Kobold2D Reference Documentation contains the class and other code references for the Kobold2D source code as well as all libraries distributed with Kobold2D, including Cocos2D, Box2D, Chipmunk, Cocos2D Extensions, Cocos3D. The Reference documentation is split into iOS and Mac references because several libraries have code features that are exclusive to a specific platform.

You can also find references to additional external learning resources for some of the libraries, Cocos2D in particular.

Frequently Asked Questions

The extensive Kobold2D FAQ answers frequently asked questions about technical and general topics.

Release Notes

If you recently installed a new version of Kobold2D you should check out the Kobold2D Release Notes to learn what's new.

Kobold2D User's Guide

- Kobold2D Requirements
- Installing Kobold2D
- Creating a Kobold2D Project
- Working with Kobold2D Projects
- Upgrading Kobold2D Projects
- Migrating a Cocos2D Project
- The Kobold2D Template Projects
- Creating a Kobold2D Template Project
- Installing a Kobold2D Template Project
- Xcode Build Locations Setup for Kobold2D

Kobold2D Requirements

Minimum Requirements for Kobold2D v2.0

- Xcode 4.2
- iOS 4.3
- Mac OS X Snow Leopard (10.6)
- ~500 MB free disk space

Minimum Requirements for Kobold2D v1.x

- Xcode 4.1
- iOS 4.0 (iOS 3.1 with Kobold2D v1.0)
- Mac OS X Snow Leopard (10.6)
- ~500 MB free disk space

Installing Kobold2D

Minimum System/Software Requirements for Kobold2D
Please check the Kobold2D Requirements before installing Kobold2D.

1. Download Kobold2D
2. Run **Kobold2D.pkg** installer

3. Follow on-screen instructions
4. Enjoy!

Kobold2D is installed to your home directory:

```
~/Kobold2D/
```

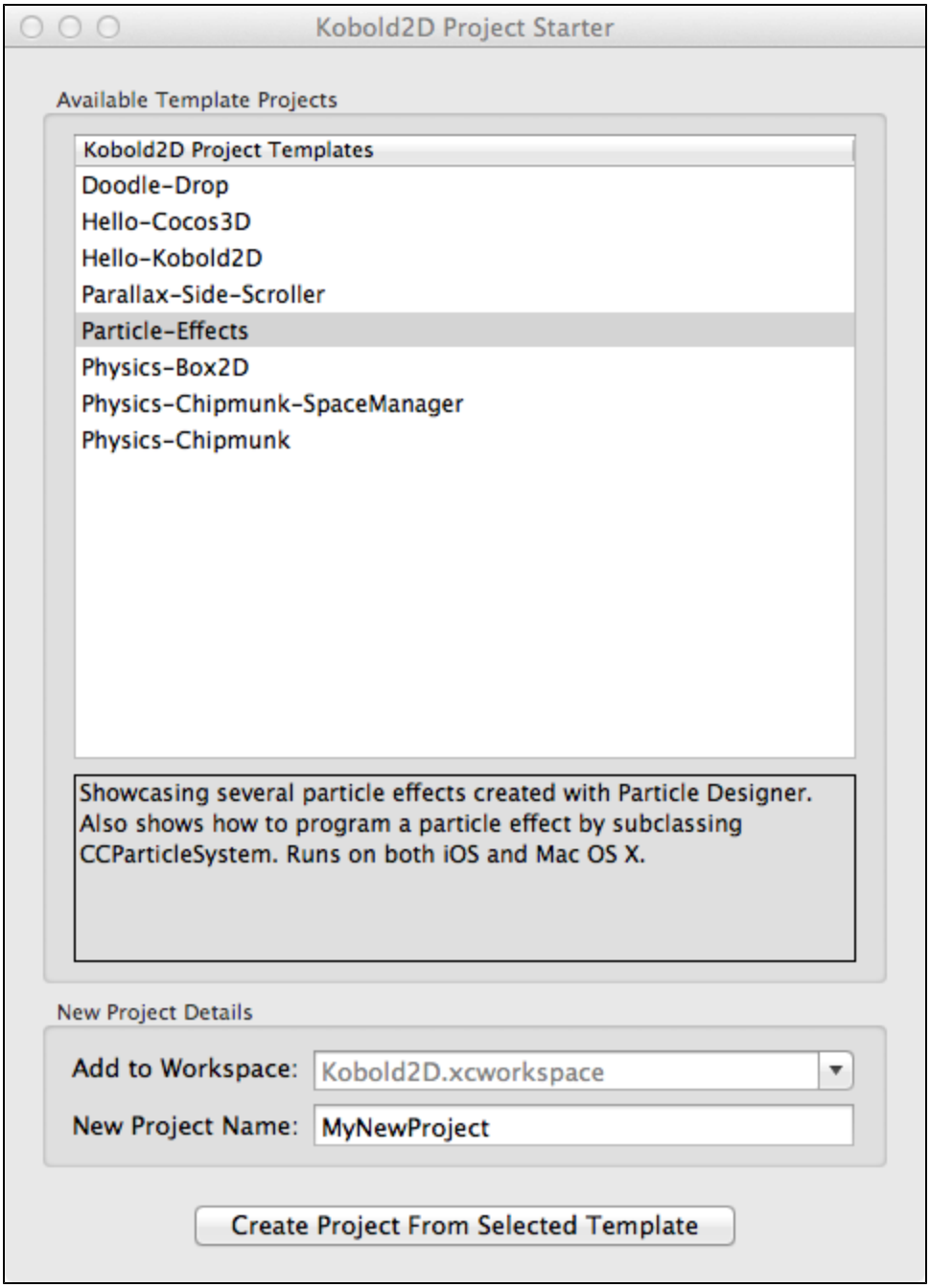
Should you need to make Kobold2D accessible to all authenticated users on the machine, you can safely move the `~/Kobold2D` folder to `/Applications/Kobold2D` or anywhere else. You may have to update folder permissions to grant other users Read & Write permissions.

Continue with Creating a Kobold2D Project.

Creating a Kobold2D Project

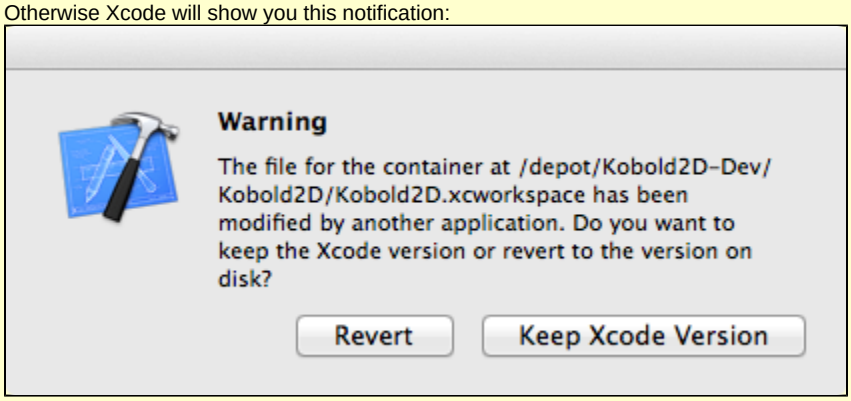
Creating a new Kobold2D project is done through the **Kobold2D Project Starter.app**. The app is located in the `~Kobold2D\Kobold2D-X.Y` folder (X.Y is the version number of Kobold2D).

1. Run **Kobold2D Project Starter.app**:



2. Select one of the provided template projects.
3. This step is optional:
    a. Select a different workspace from the **Add to Workspace** list *or*
    b. Enter the name of a new workspace in the **Add to Workspace** text field.
4. Enter the name for your new project.
5. Click **Create Project From Selected Template** to have the project and, if necessary, the workspace created for you.
6. Xcode will open the selected workspace which contains the newly created project.
7. Select your project and a target platform from the scheme dropdown and click the Run button.

Close open workspaces in Xcode
It is recommended to close the workspace in Xcode (or close Xcode altogether) before adding a new project to that workspace. Otherwise Xcode will show you this notification:


Warning
The file for the container at /depot/Kobold2D-Dev/Kobold2D/Kobold2D.xcworkspace has been modified by another application. Do you want to keep the Xcode version or revert to the version on disk?
Revert
Keep Xcode Version

If you click **Keep Xcode Version** your newly created project will not appear in the Kobold2D workspace because Xcode reverts the changes done by the Project Starter tool. If you click **Revert** your new project will be correctly added to the workspace, but you risk losing unsaved changes.

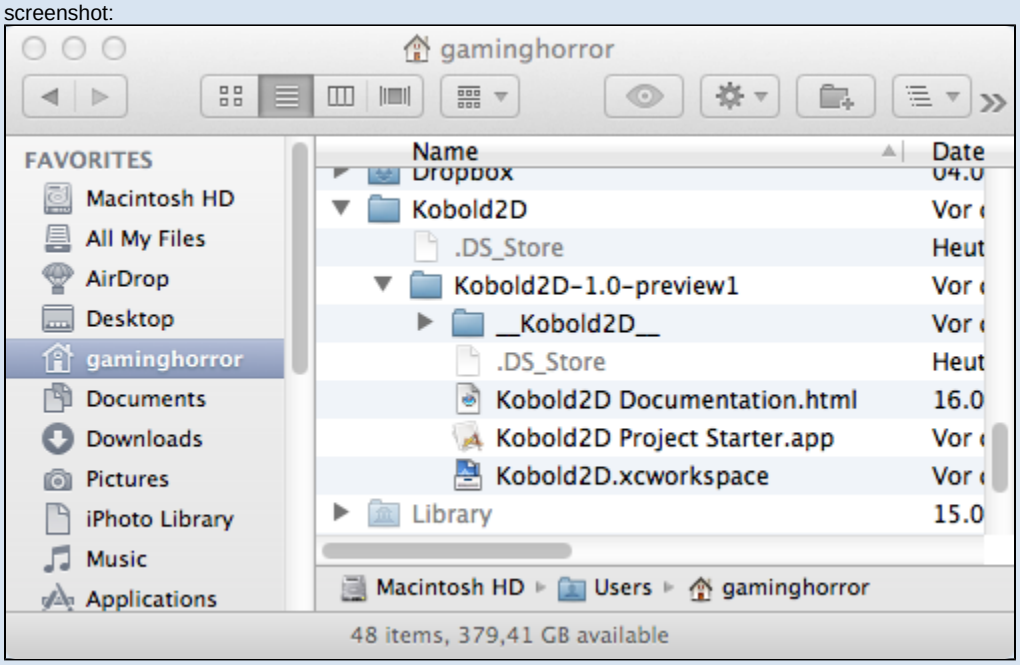
No Xcode Project Templates
Due to technical limitations of the built-in Xcode 4 project template format Kobold2D does not provide project templates accessible through Xcode's **File -> New -> New Project...** dialog.

Continue with Working with Kobold2D Projects.

Working with Kobold2D Projects

The Kobold2D Folders

Looking at the Kobold2D folders after installation, you should see folder contents similar to the ones in this example Finder screenshot:



The `__Kobold2D__` folder contains all Kobold2D files. Unless you know what you are doing or even plan to contribute to Kobold2D it is highly recommended to leave all files in the `__Kobold2D__` folder and its subfolders unchanged.

The Kobold2D Workspace

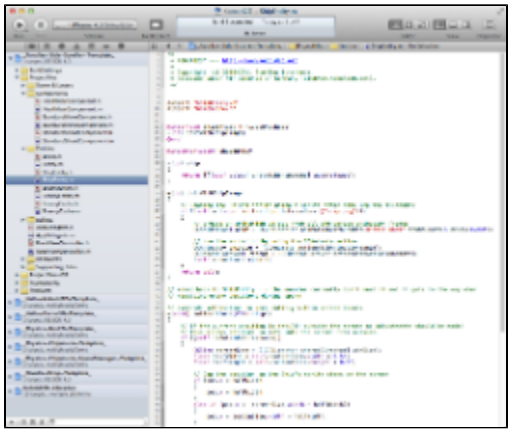
In each `~/Kobold2D/Kobold2D-X.Y` is a `Kobold2D.xcworkspace` (referred to simply as the "Kobold2D workspace") which you can open with Xcode 4 or newer. By default the Kobold2D workspace will contain all of your projects. You can also create your own workspace.

> Do not open the .xcodeproj of a Kobold2D project - this will fail to build. You always have to open a project's .xcworkspace file, which in addition to your project also contains the mandatory Kobold2D-Libraries project.

Your Kobold2D Projects

After installing Kobold2D the Kobold2D workspace contains only the Kobold2D libraries project. You need to create a new project from a template project before you can work with the Kobold2D workspace.

Here's a screenshot of the Kobold2D workspace from the development branch, which contains all template projects:

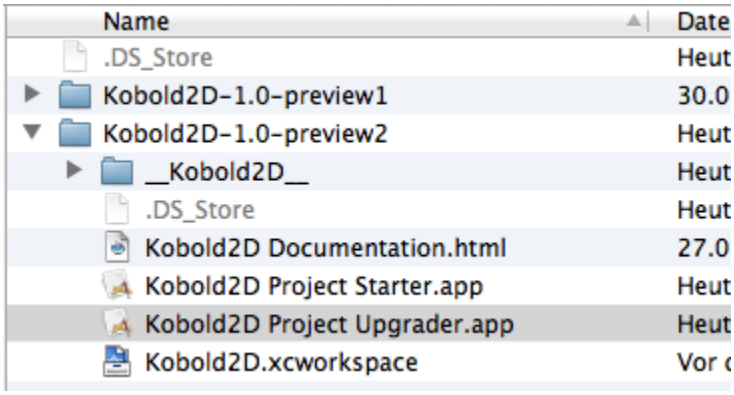


Each project in the Kobold2D workspace has its own scheme that allows you to switch projects quickly and easily. Once you get to work on your second Kobold2D project you'll enjoy being able to quickly look up your previous project(s) files.

Upgrading Kobold2D Projects

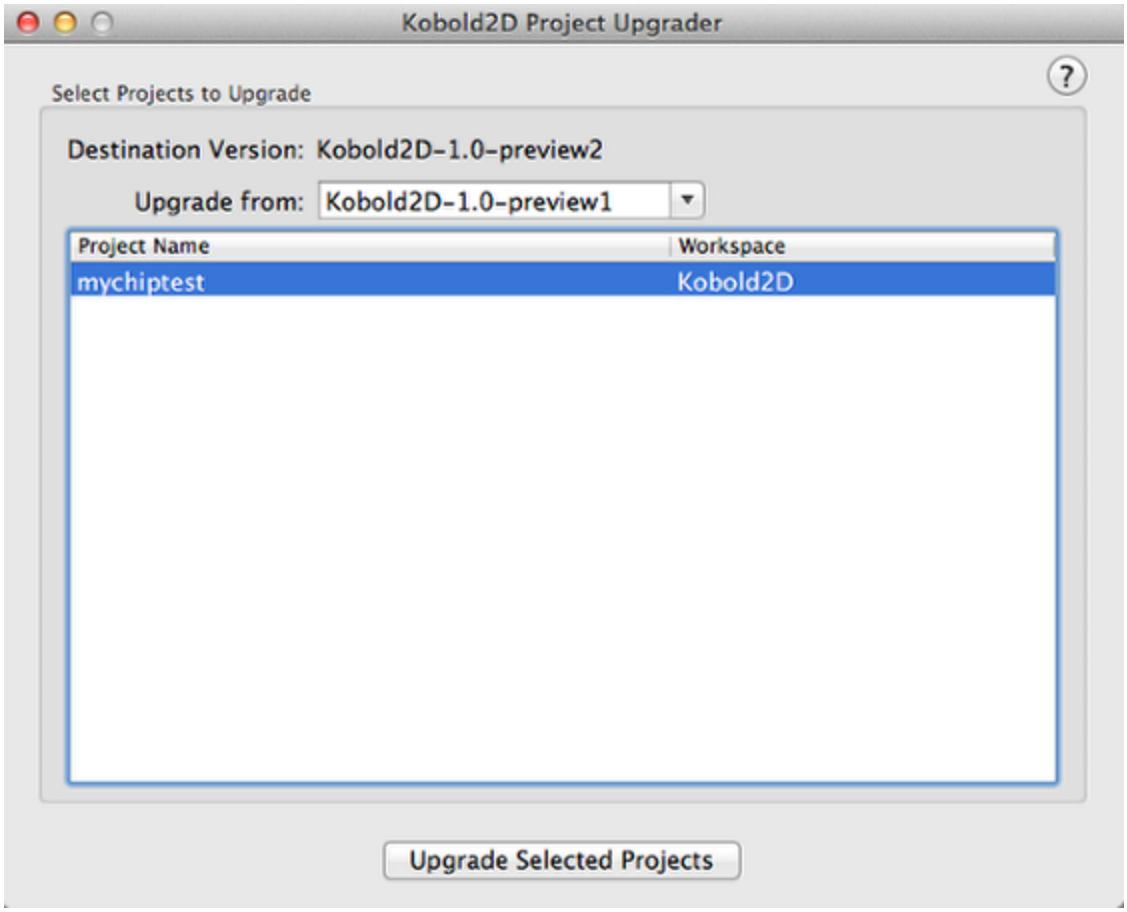
> This guide explains how you can upgrade (or move) an existing Kobold2D project to a newer version of Kobold2D.

Run the Kobold2D Project Upgrader.app



After installing a new Kobold2D version, open that version's root folder and run the **Kobold2D Project Upgrader.app**.

Select the projects you want to upgrade



The **Kobold2D Project Upgrader.app** will scan the other Kobold2D folders for upgradeable projects and show them in a list.

By default all upgradeable projects are already highlighted. You can also select only individual projects in the same way you select files in Finder: Shift+Mouseclick to select a range of projects, or Command+Mouseclick to select or deselect individual projects.

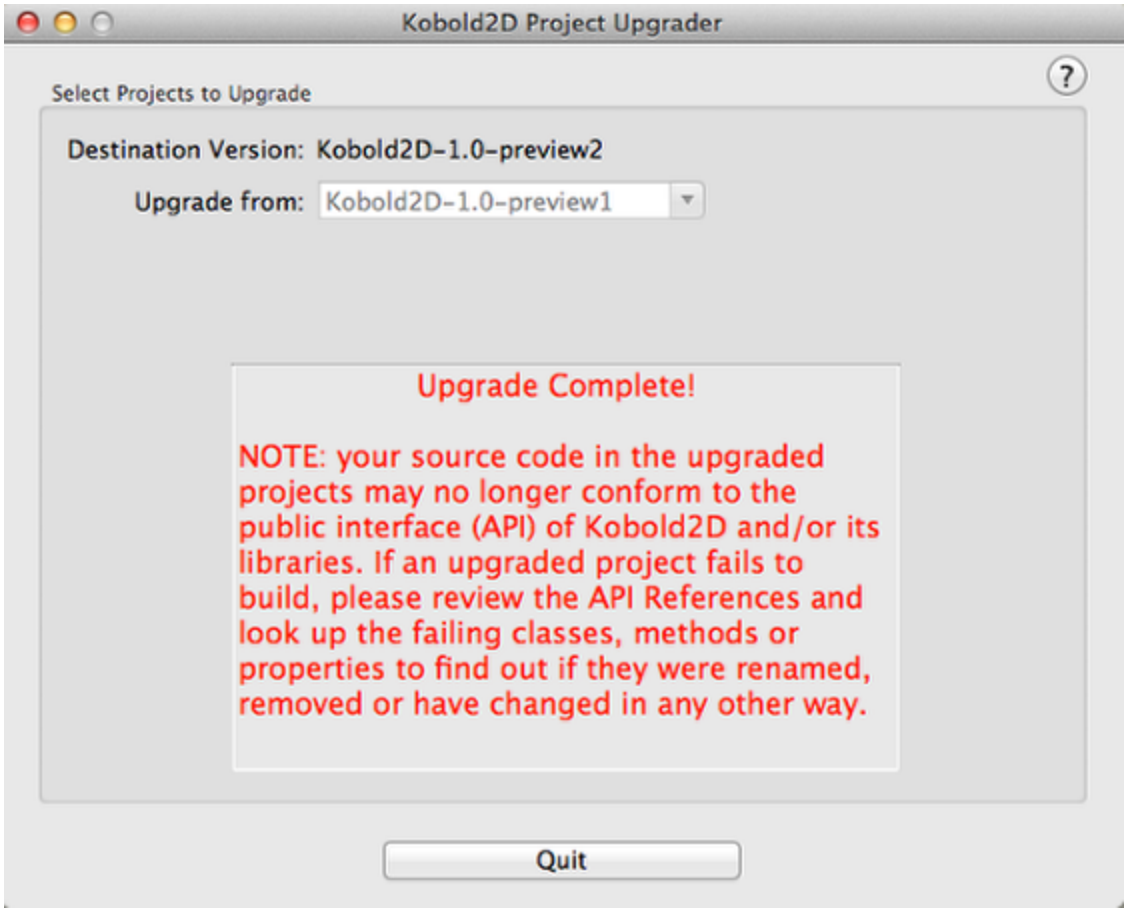
The **Upgrade from** setting allows you to upgrade Kobold2D projects from other, possibly earlier versions of Kobold2D (if available). By default the projects from the second to last installed Kobold2D version are shown. Normally you don't need to change this setting.

If a project is not in the list...
The **Kobold2D Project Upgrader.app** tries to ensure that you don't accidentally overwrite already existing projects. For that reason not all projects may be listed. In particular projects for which a folder with the same name already exists in the destination version's folder will not be shown to prevent accidentally overwriting that project.

In a few cases this may not be an already upgraded project. Assuming you have already created a project with the same name (eg `test`) in the preview 2 version and then want to update the older, different but also named `test` project from preview 1, you will first have to rename or delete either one of these two `test` projects before you can upgrade the old `test` project.

Click "Upgrade Selected Projects"

Close Xcode!
It is recommended to close Xcode before clicking the **Upgrade Selected Projects** button. At the very least you should close the workspaces and projects that you are going to upgrade.


Kobold2D Project Upgrader
Select Projects to Upgrade
Destination Version: Kobold2D-1.0-preview2
Upgrade from: Kobold2D-1.0-preview1
Upgrade Complete!
NOTE: your source code in the upgraded projects may no longer conform to the public interface (API) of Kobold2D and/or its libraries. If an upgraded project fails to build, please review the API References and look up the failing classes, methods or properties to find out if they were renamed, removed or have changed in any other way.
Quit

The upgrader tool will copy the project folder and its contents over to the destination version, in this case `Kobold2D-1.0-preview2`. It will also upgrade the project's workspaces.

The informational message is a reminder that changes to the API may cause build errors in the upgraded projects, which are not related to the upgrade process itself. The **Kobold2D Project Upgrade.app** can not modify your source code to make it compatible with the latest APIs. That is always a manual process, but in most cases the necessary changes are minimal. Please refer to the Kobold2D Release Notes for information on API changes.

You can now open the workspaces in the destination Kobold2D version and continue working.

> **Be aware of which workspace version Xcode opens!**
> Keep in mind that Xcode automatically re-opens the most recently used workspaces and projects. If you do not manually open the corresponding workspace(s) from the **latest Kobold2D version** you will **continue to work with the second to last version of Kobold2D**, which is probably not what you want.

Migrating a Cocos2D Project

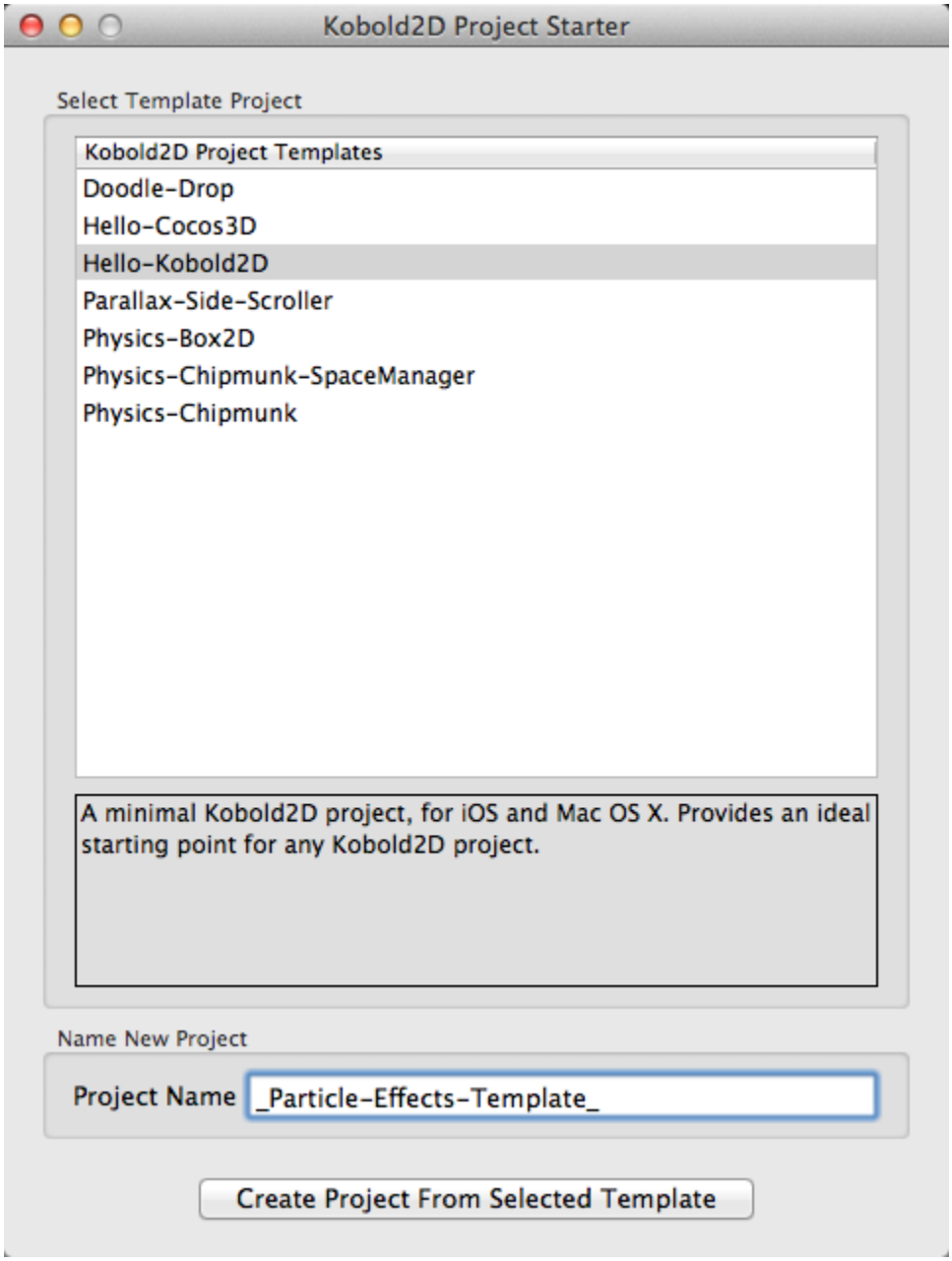
This guide explains by example how an existing Cocos2D project can be converted to a Kobold2D project. In this step by step guide the project **ParticleEffects03** from the Learn Cocos2D book will be migrated to Kobold2D.

This is a general guide to explain the process of converting an existing Cocos2D project to Kobold2D. This guide can not offer a solution to every possibly required change to successfully migrate every project. Each project is different and may require custom modifications or experience issues not outlined in this guide. Please use the Kobold2D Forum if you need additional help on migrating a project to Kobold2D.

> If your project is nearing completion it may be better to finish that project as is. You won't benefit much from converting an almost complete project to Kobold2D and you risk running into subtle and time-consuming issues that might delay the project's release. Instead just start your next project straight as a Kobold2D project.

⚠ As you migrate from your Cocos2D project to a Kobold2D project, be aware that the third party libraries provided by Kobold2D (including but not limited to: cocos2d-iphone, cocos3d, Box2D, Chipmunk, etc) may be of a different version. Be prepared to adapt your code accordingly.

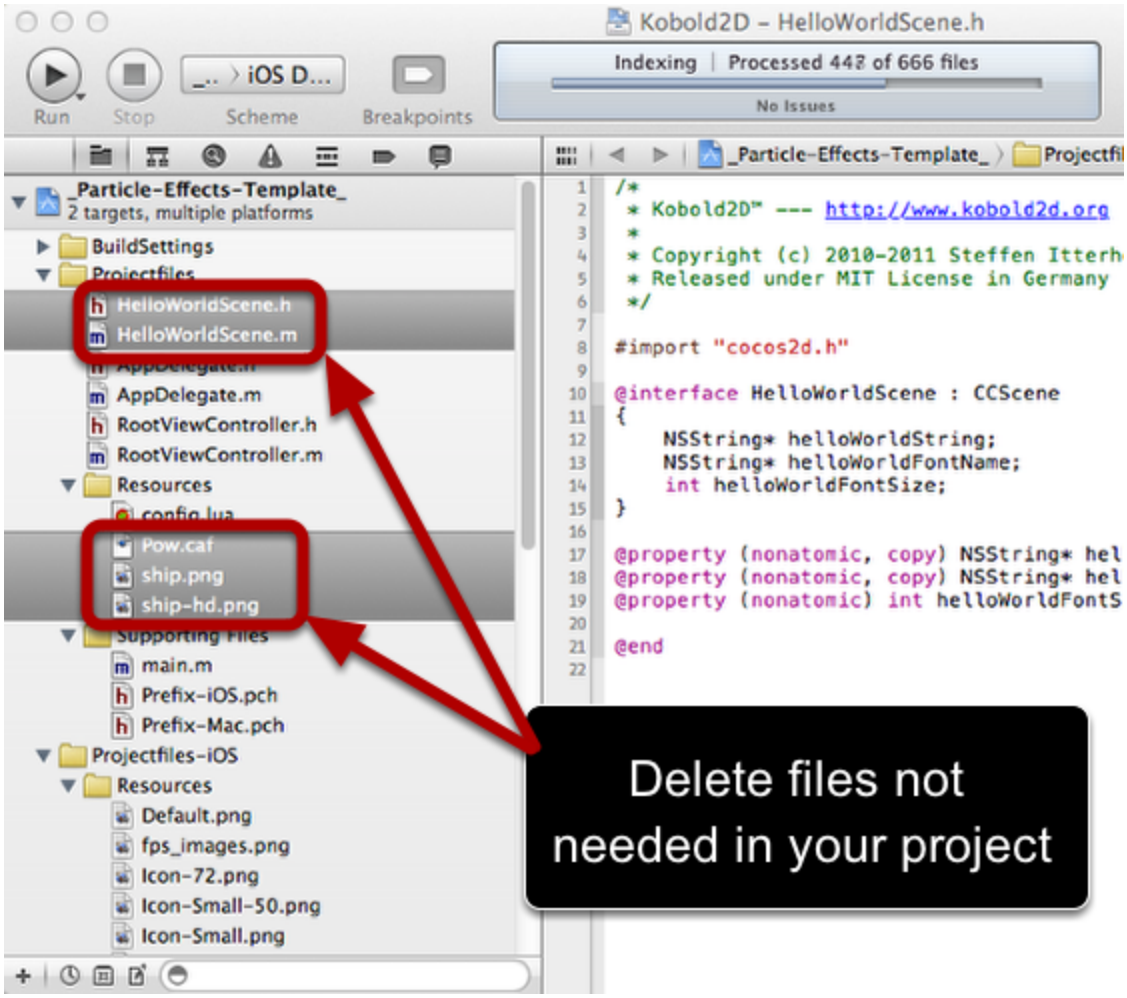
Create a Kobold2D Project



The first step is to create a new Kobold2D project from one of the templates. It is recommended to use the Empty-Project template (screenshot pictures the Hello Kobold2D template) even if your project uses a physics engine. All Kobold2D project templates are already setup for both physics engines.

The new project is named _**Particle-Effects-Template**_ because the goal is to turn it into a Kobold2D template project. You should use an appropriate name for your new project.

Cleanup the new Kobold2D Project

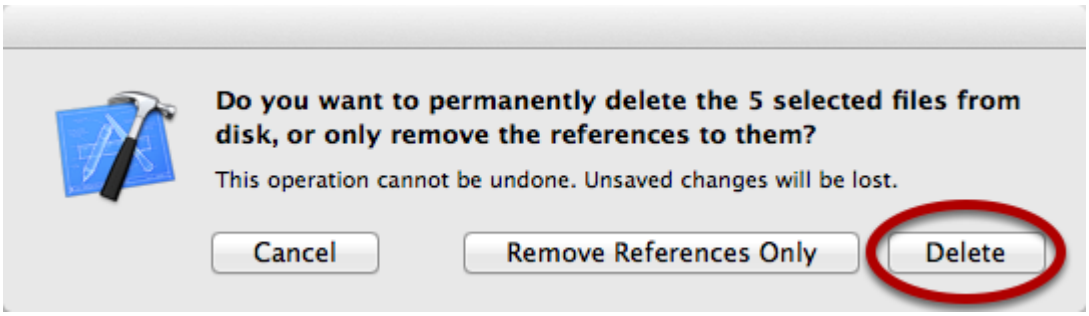


There are a few files which won't be needed by your project, depending on which project you use as a starting point. Hence the recommendation to pick a simple project, preferably the Empty-Project template.

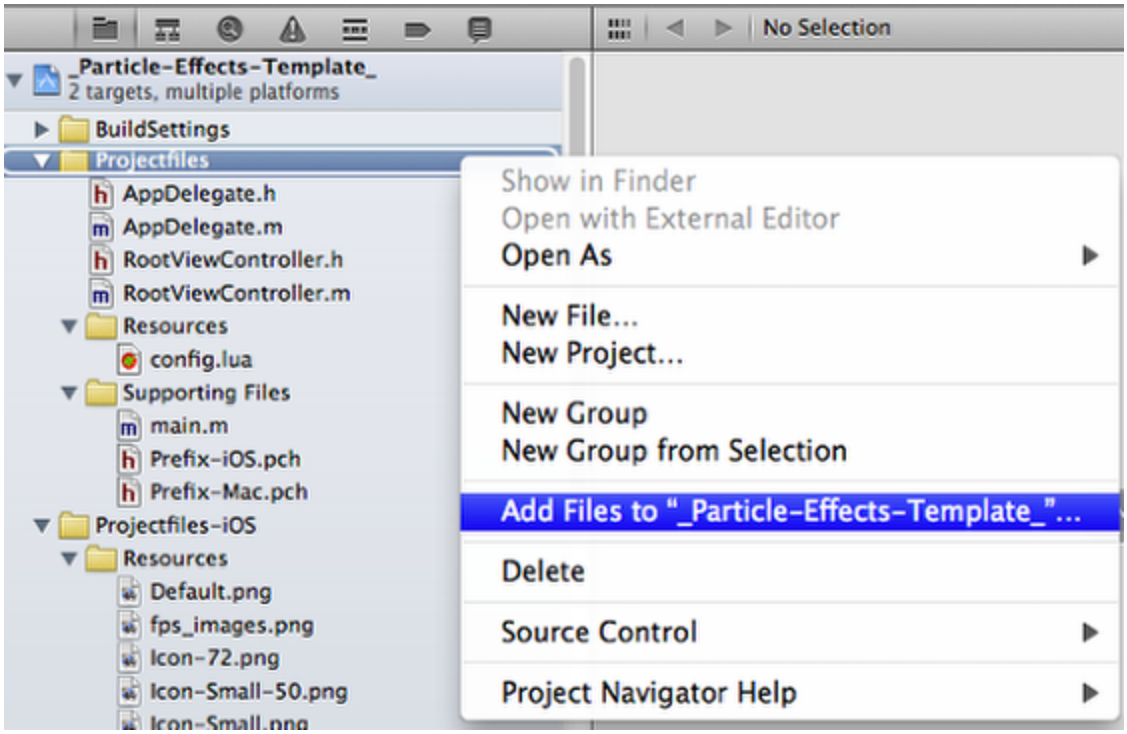
For example classes like the HelloWorldScene class and several resource files (Pow.caf, ship.png, ship-hd.png) will not be needed in your converted project.

Keep config.lua
Do not delete the file **config.lua**! Kobold2D projects require it.

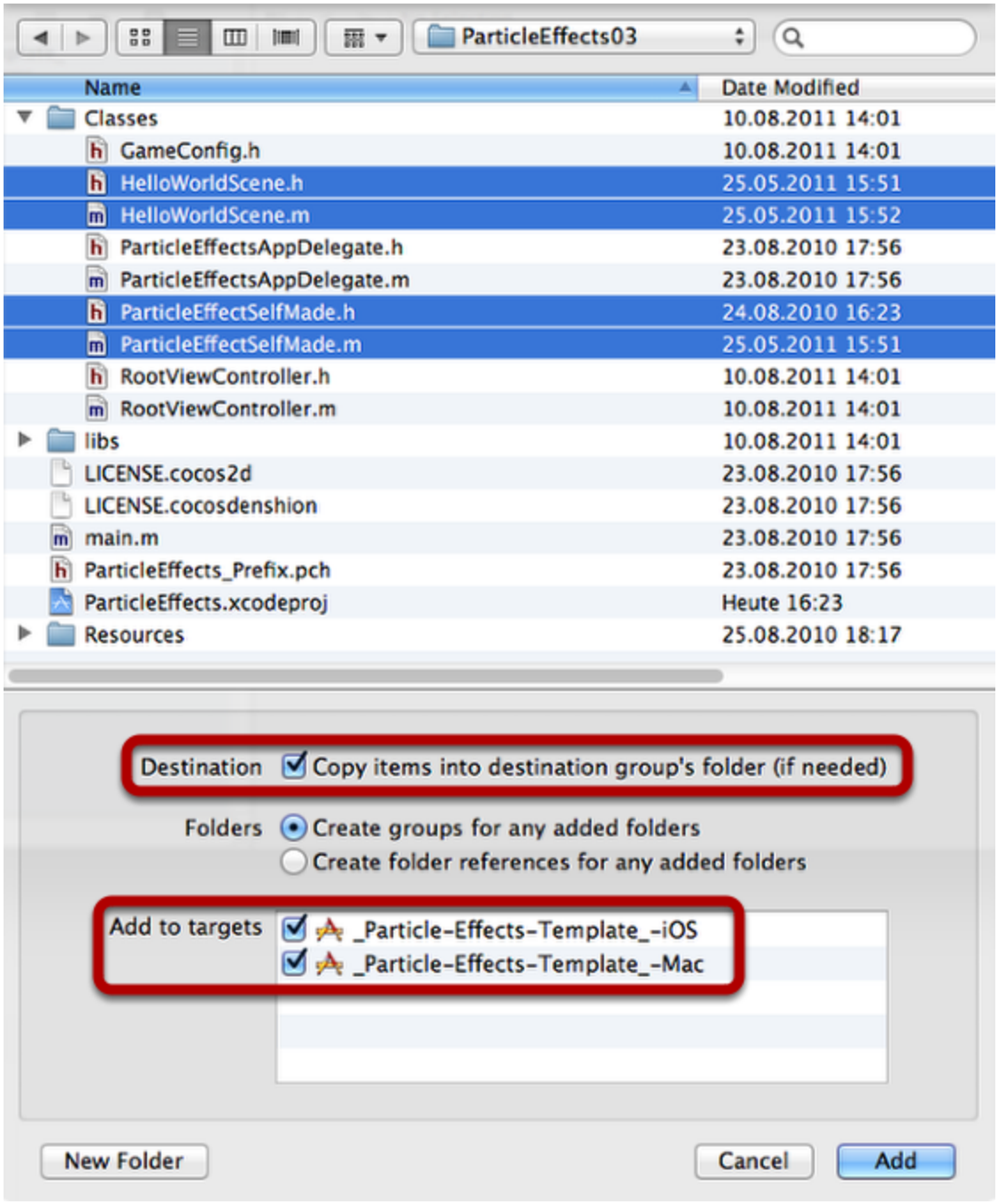
When removing unneeded source code and resource files you will want to **Delete** these files to also remove them from the file system:


Do you want to permanently delete the 5 selected files from disk, or only remove the references to them?
This operation cannot be undone. Unsaved changes will be lost.
Cancel
Remove References Only
Delete

Add your project's source code



Select the **Projectfiles** group, right-click or option-click to bring up the menu. Then select **Add Files to "projectname"...**. This will bring up the following Add Files dialog:



Locate your original project's source code files. Be sure to select all of your source code, but only the source code that you wrote.

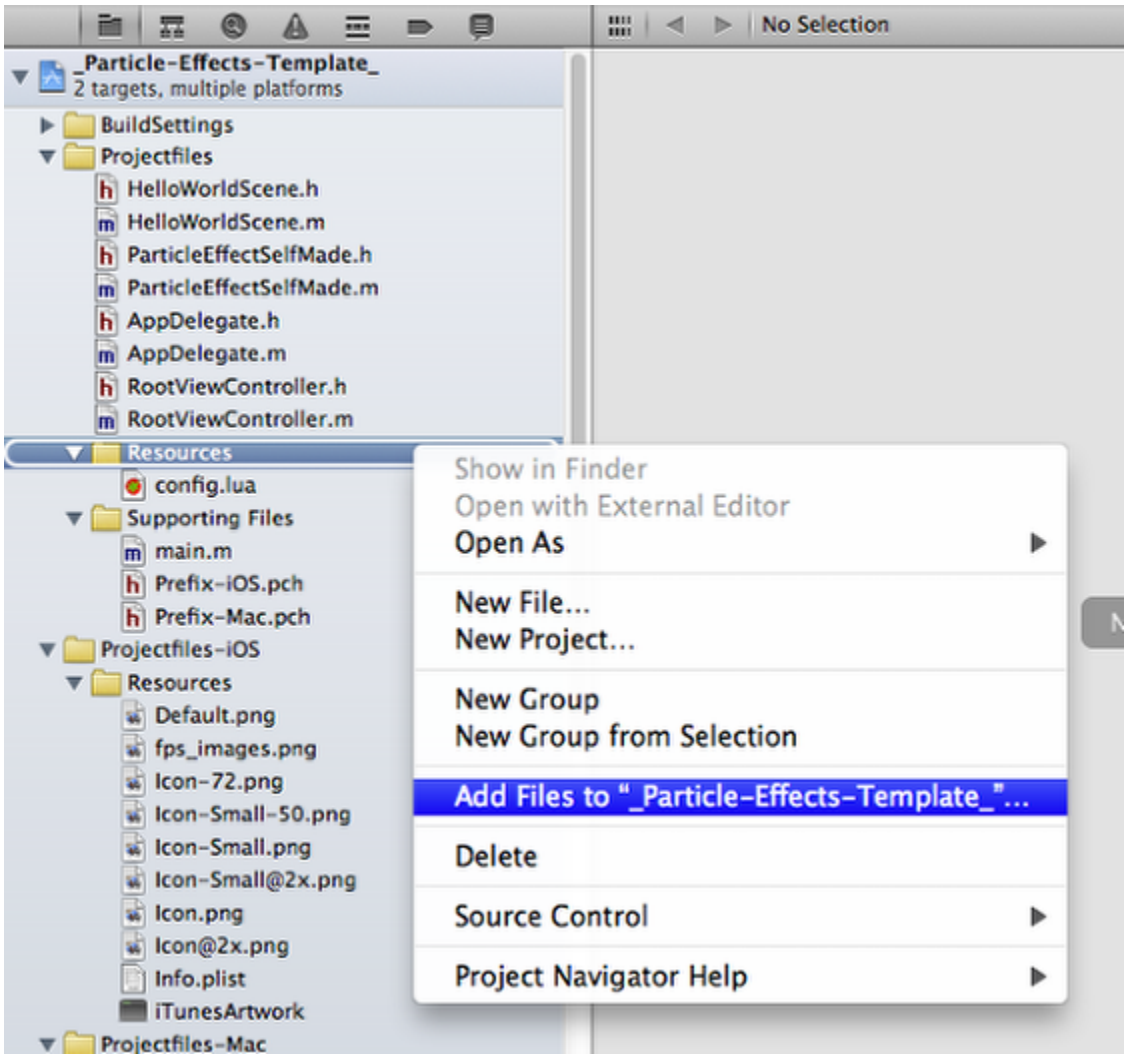
In particular, make sure you **do not include** the following files:

- GameConfig.h
- (ProjectName)AppDelegate.*
- RootViewController.*
- main.m
- (ProjectName)_Prefix.pch
- any library source code, such as: cocos2d, box2d, chipmunk, etc. source code files

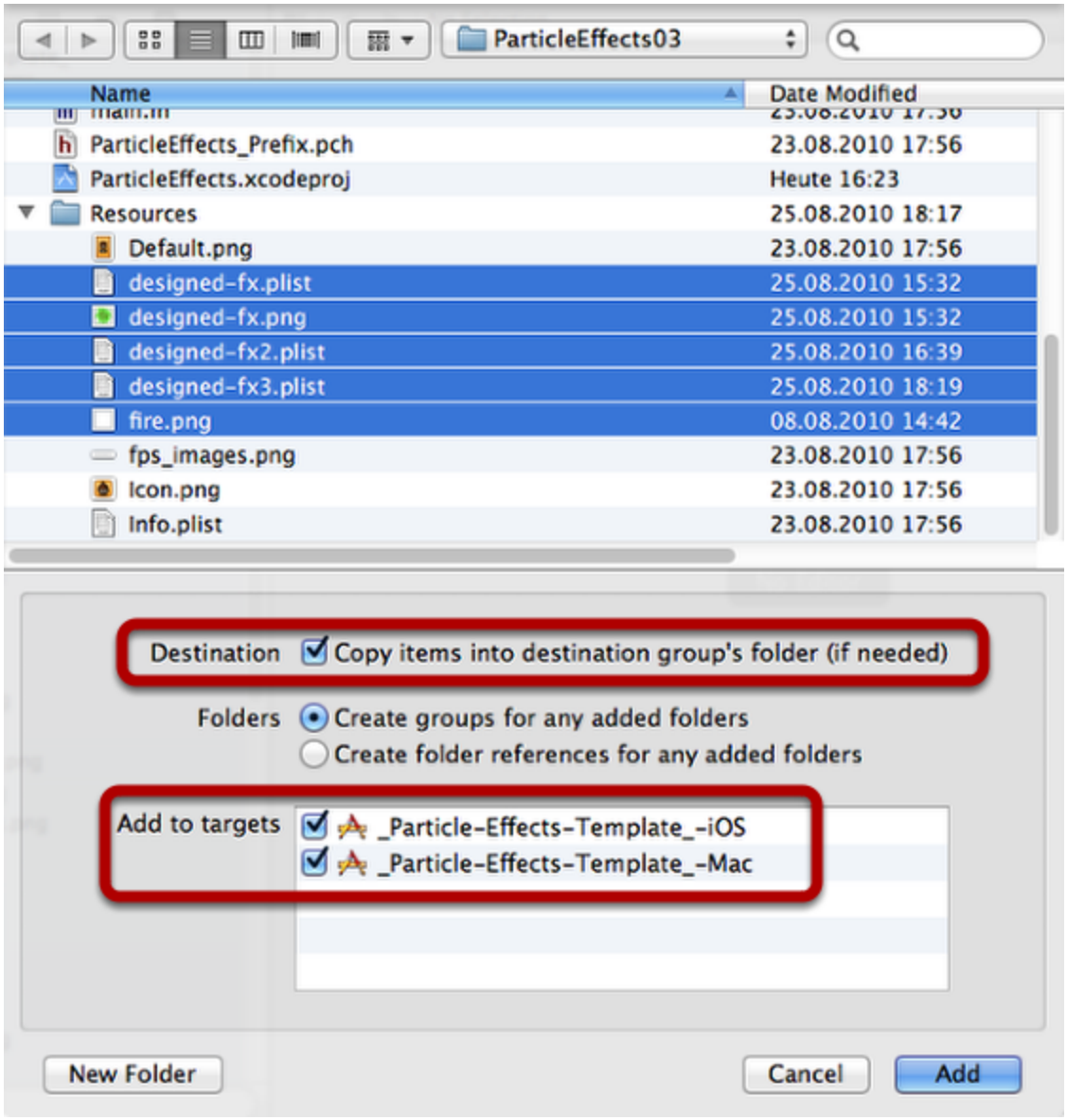
In case you accidentally added one of the above files you should select and delete them from the Kobold2D project. Just make sure you delete those you added and not the ones that may have already existed in the Kobold2D project.

Before clicking **Add** make sure you've selected the checkbox **Copy items into destination group's folder (if needed)** and the corresponding targets that you want to support (if in doubt select both).

Add your project's resource files



Now select the Resources group and option-click or right-click. Once again select **Add Files to "ProjectName"...** to bring up the Add Files dialog again, this time for adding your project's resource files:



Select the resource files that are unique to your project, eg the ones that you created and added.

Do not include the standard resource files! In particular, make sure you **do not include** the following files:

- Info.plist
- Default.png, Icon.png or any variant (Default@2x.png, Icon@2x.png, etc.)

In case you accidentally added one of the above files you should select and delete them from the Kobold2D project. Just make sure you delete those you added and not the ones that may have already existed in the Kobold2D project.

If you have modified any of the default resource files above, make a note (preferably not a mental note as you'll likely forget - you are human, right?). That note should read:
Remember to copy my project's modified standard resource files over the ones provided in the Kobold2D project, using Finder.
Also remember to check the Info.plist file for any modifications/additions that I need to copy from my original project to the new Kobold2D project's Info.plist file.

About Info.plist
You should be aware that the Hello Kobold2D project contains two Info.plist files, one for iOS apps and one for Mac OS X apps. If you plan to support both platforms you may need to modify both. If you don't, make sure you modify the correct one. Platform-specific resource files are located in the **Projectfiles-iOS** respectively **Projectfiles-Mac** folders.

Before clicking **Add** make sure the Add Files settings are correct. The checkbox **Copy items into destination group's folder (if needed)** must be checked as well as the corresponding targets that you want to support (if in doubt select both).

Modify the config.lua

Particle-Effects-Template › Projectfiles › Resources › config.lua › No Selection
Your first scene's class name
Need multi-touch?
Do you need custom render settings?
Retina enabled?
Orientation & Autorotation?
Need iAd banners?
Mac support?

```
local config =
{
    KKStartupConfig =
    {
        -- load first scene from a class with this name
        FirstSceneClassName = "HelloWorldScene",

        -- set the director type, and the fallback in case the first isn't available
        DirectorType = DirectorType.DisplayLink,
        DirectorTypeFallback = DirectorType.NSTimer,

        MaxFrameRate = 60,
        DisplayFPS = YES,

        EnableUserInteraction = YES,
        EnableMultiTouch = NO,

        DefaultTexturePixelFormat = TexturePixelFormat.RGBA8888,
        GLViewColorFormat = GLViewColorFormat.RGB565,
        GLViewDepthFormat = GLViewDepthFormat.DepthNone,
        GLViewPreserveBackBuffer = NO,
        GLViewMultiSampling = NO,
        GLViewNumberOfSamples = 0,

        Enable2DProjection = NO,
        EnableRetinaDisplaySupport = YES,

        DeviceOrientation = DeviceOrientation.Portrait,
        AutorotationType = Autorotation.UIViewController,
        ShouldAutorotateToLandscapeOrientations = YES,
        ShouldAutorotateToPortraitOrientations = YES,
        AllowAutorotateOnFirstAndSecondGenerationDevices = NO,

        -- iAd setup
        -- Note: for iAd to support autorotation you should use: AutorotationType =
        ViewController
        EnableAdBanner = YES,
        LoadOnlyPortraitBanners = NO,
        LoadOnlyLandscapeBanners = NO,
        PlaceBannerOnBottom = NO,

        AutoScale = YES,
        AcceptsMouseMovedEvents = NO,
        WindowFrame = RectMake(1024-640, 768-480, 640, 480),
    },

    -- you can create your own config sections using the same mechanism and use KKConfig to
       access the parameters
    -- or use the KKConfig injectPropertiesFromKeyPath method
    HelloWorldSettings =
    {
        HelloWorldString = "Hello Kobold2D!",
        HelloWorldFontName = "Marker Felt",
        HelloWorldFontSize = 50,
    },
}
```

Now open the file **config.lua** in the new Kobold2D project. You will find it in the Projectfiles/Resources group. The above screenshot gives you hints as to what you may want to or need to change.

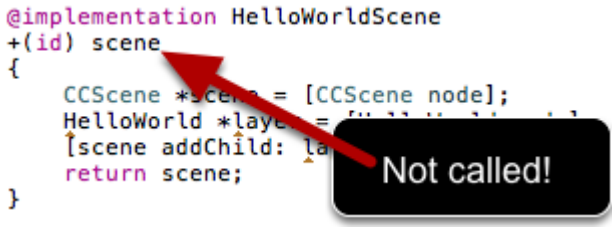
At the very least you will have to change the FirstSceneClassName parameter to the name of the CCScene or CCLayer class that Kobold2D should load as the very first scene. You may also want to verify that the initial device orientation and autorotation match those of the Cocos2D project.

FirstSceneClassName is a class name, not file name!
Note that this parameter should indicate the class name, not the file name. In the case of the particles project the file names of the first scene was HelloWorldScene.h respectively .m but the class was actually called just **HelloWorld**.

Do not remove entries from **config.lua** which are seemingly unneeded. Kobold2D may rely on those, and you may later want to be able to change some of these settings without having to look them up in the Kobold2D documentation.

If you are unsure which settings were used by your Cocos2D project, have a look at that project's AppDelegate.m and GameConfig.h files. If you don't remember making any changes to these files the defaults provided by Kobold2D should work fine. If you do encounter strange issues in the Kobold2D version of your project, such as multi-touch not working or performance degradation, it is most likely caused by different startup settings. Please refer to the Config.lua Settings Reference for more information.

Changes to your "first scene" class



If you allow Kobold2D to run your first scene via the config.lua setting **FirstSceneClassName** then the standard class method provided by Cocos2D project templates, namely the method `+(id) scene` is no longer being called. If the scene method contains just the default code adding a layer to the scene, you will probably not notice a difference. But if you have custom code in that method, you have these options:

1. rename the method from `(id) scene` to (id) node in the implementation and interface
2. you can also move the `+(id) scene` code to layer's init method
3. call your first scene manually in the AppDelegate method `-(void) initializationComplete`

Option #1 is the simplest solution. It should work in most cases flawlessly. But if not you might want to try the other options.

Option #2 is usually trivial, if all you did in the scene method was adding more layers. Those layers can also be added in the init method of the same class. If that class is itself a layer, try adding the new layers to `[self.parent addChild:myLayer]` instead of directly to self (the HelloWorld class in this case).

Option #3 is also trivial and simply overrides Kobold2D's way of running the first scene by reverting to the old-school, Cocos2D way to launch the first scene. The code added to the AppDelegate class would look similar to this and ensures that the `+(id) scene` method is called as usual:

```
-(void) initializationComplete
{
 [[CCDirector sharedDirector] runWithScene:[MyFirstScene scene]];
}
```

Other Alterations

Some projects may require additional changes or manually migrating code. That is where you should carefully test each step in the "migratory process" and where we'll find it difficult to provide support, as every project is different. But feel free to ask for help in the Kobold2D Forum.

Take Small Steps
If you have to tread unpaved territory when migrating your project it helps to first get your project up and running, if incomplete. Then migrate custom code step-by-step in small bunches to the Kobold2D project, if necessary commenting out some code to make it work. Always test each step. Repeat this process to slowly but surely and consciously get closer to the final, working project.

Migrating AppDelegate & RootViewController changes

In particular if you have modified the AppDelegate or RootViewController classes you will have to migrate these changes step by step to the Kobold2D project. Just be sure to call the super implementations for any overridden methods in those classes, as Kobold2D implements default AppDelegate and RootViewController classes which perform default operations, such as properly enabling and performing autorotation based on the config.lua settings.

Migrating Default Resources

Remember the non-mental notes you should keep? Now is the time to compare the Info.plist files of the two projects and merging any changes to the ones in the Kobold2D project.

Likewise, any modified default image (Default.png, Icon.png) you should now simply copy over the existing ones in the Kobold2D project's Resources folders, using Finder. You can then verify in Xcode by selecting these files that they have been correctly updated.

Migrating a third party library not provided by Kobold2D

This depends on how you've added that library to your project. If it was, like Cocos2D, simply added to your project by dropping the library's source code files to your project's target, you can just treat that library's code as if it were your own.

If the code is compiled as a static library you may have to add that particular target (or one for each supported platform) to your Kobold2D project.

Let us know!
If you think the library you're using would be great to have in Kobold2D, and provided that library is free to distribute and open source, you might want to request that library being added to Kobold2D. You can do so in the Kobold2D Forum.

Warnings and errors that weren't there before

Migrating code to a Kobold2D project can give you warnings or errors that weren't there before. It's possible that the code still works but isn't 100% conforming to good coding practices. For example, the Xcode default build settings will not warn you about possibly undeclared selectors, instead your app will just crash while it is running. Kobold2D has this warning enabled by default, and also turns all warnings into errors since warnings should be taken serious.

The simplest fix is to find the offending compiler warning setting and to turn it off (or on) in your project's build settings. But you might want to take the opportunity and check if the new warnings/errors indicate potential bugs in your project.

Removing unsupported platforms

If your project does not support a particular platform, be it iOS or Mac, then you may want to remove:

- the platform's target
- the platform's build scheme
- platform-specific resource files (see Projectfiles-iOS respectively Projectfiles-Mac)

You may want to consider keeping the unnecessary platform specific targets if there's even a slight possibility that you might want to go cross-platform eventually. Adding a platform specific target back in isn't as straightforward as it may seem.

The Kobold2D Template Projects

Learn iOS Game Development

The Learn Cocos2D book symbol indicates in which chapter or article you can learn more about each template project.

Basic Template Projects

Hello Kobold2D

iOS

Mac OS

The standard template project for Kobold2D is the successor of the "Hello Cocos2D" template. It showcases Retina support, config.lua settings, OS and device detection, iAd banners, actions, built-in and custom font labels and sound.

Chapter 16

Use this template as a starting point if you want to start a Kobold2D project from scratch.

Hello Cocos3D

iOS

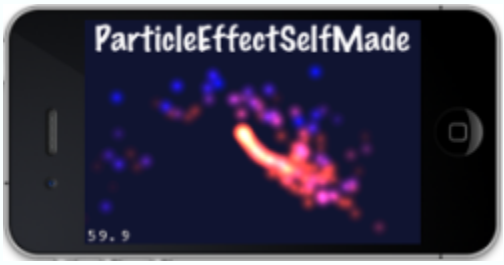
iOS

This template is similar to *Hello Kobold2D* and showcases the use of Cocos3D, mixed with regular Cocos2D nodes. Notice that Cocos2D nodes can be drawn in front of or behind 3D objects. Cocos3D currently does not support Mac builds.

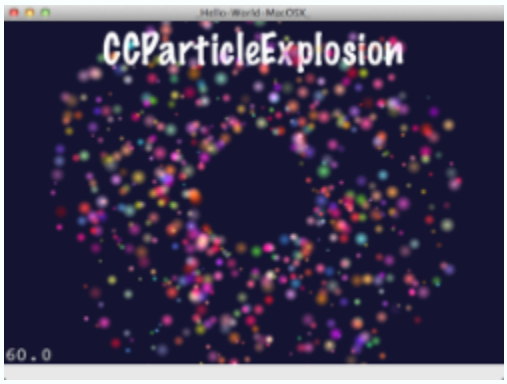
Chapter 16

Use this template as a starting point if you plan to work with Cocos3D. It has the required line `-force_load libcocos3d.a` in `BuildSettings-iOS.xcconfig` already set.

Particle Effects



iOS



Mac OS

This project shows you how to create particle effects by subclassing the CCParticleSystem class. It also displayed particle effects created with Particle Designer as well as Cocos2D's built-in particle effects.

Chapter 9

User Input

iOS

Mac OS

Illustrates how to use the KKInput class which wraps user input into a collection of convenience method. The main difference being that KKInput allows you to poll the input state at any time from any class without having to register to receive input events. KKInput takes the grunt work out of input processing, giving you the ability to easily test for double-clicks, scroll wheel & mouse position changes, key states with or without modifiers, and more.

Exclusive Kobold2D Feature!

Advanced Template Projects

Physics Box2D / Physics Chipmunk / Physics Chipmunk SpaceManager

iOS

Mac OS

The three physics template projects in Kobold2D all implement the same physics demo: adding dynamic bodies (boxes), collision callbacks, and the screen/window acting as an impenetrable barrier. A static body is used to attach three dynamic bodies which are connected via Revolute (Box2D) / Pivot (Chipmunk) joints to create a swinging "rope".

Chapter 12

If you're unsure which physics engine is right for you, check out these templates and use the one which you find easiest to work with. You might also want to read the FAQ entry: Which Physics Engine is the best?

Orthogonal Tilemap

iOS

Mac OS

The orthogonal tilemap project shows how to load a TMX tilemap created with the Tiled Map Editor. It demonstrates conversion of touches to tile coordinates, inspecting tile properties, the use of the Tiled Object Layer for trigger areas and scrolling of the tilemap.

Chapter 10

Isometric Tilemap

iOS

Mac OS

The isometric tilemap project shows how to load an isometric TMX tilemap created with the Tiled Map Editor. The isometric tilemap is drawn with depth buffering and 2D projection for correct Z ordering of tiles. The project demonstrates conversion of touches to tile coordinates, inspecting tile properties, the use of the Tiled Object Layer for trigger areas and scrolling of the tilemap. It includes a controllable player character. On Mac OS it illustrates how to process keyboard events.

Chapter 11

Sprite Performance

iOS

Mac OS

This project illustrates how to improve rendering performance by replacing sprites created from individual files to sprites created from sprite frame provided by a PVR compressed texture atlas. It also shows how to use CCSpriteBatchNode to improve performance even further.

Blog Post

This project can also be used for benchmarking sprite rendering performance of various sizes and image formats.

Cocos2D With UIKit Views

iOS

Mac version currently has unresolved issues. See comments in AppDelegate.m.

Mac OS

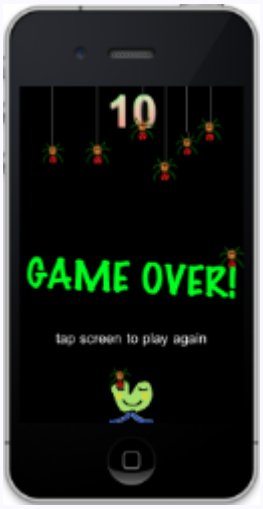
This project shows you how to add Cocoa views to Cocos2D. In particular the project not only adds views on top of Cocos2D but also underneath (transparent OpenGL view). It also correctly performs hit testing on views to emulate the responder chain for Cocos2D nodes. This is achieved by enabling `EnableGLViewNodeHitTesting` in config.lua. The user can interact with views in the foreground, cocos2d nodes in the OpenGL view, and views behind the cocos2d OpenGL view.

Chapter 15

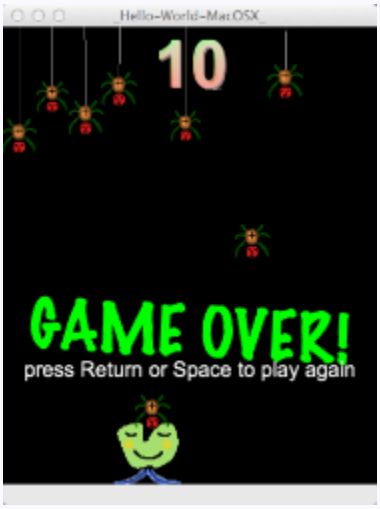
The Learn Cocos2D (2nd Edition) book Chapter 15 includes instructions on how to add Cocos2D to an existing UIKit app.

Game Template Projects

Doodle Drop



iOS



Mac OS

Doodle Drop is a fully playable game in which you attempt to avoid the dropping spiders by tilting your device. The game showcases accelerometer input (keyboard input on Mac), sound, music, a game over screen, a custom loading screen, score counting wth a bitmap font label, object pooling to avoid frequent alloc/dealloc cycles and advanced use of easing, rendering lines, simple collision testing, sequence and call func actions.

Chapter 4

Parallax Side-Scroller

iOS

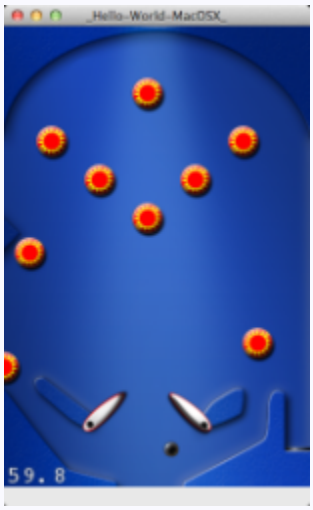
(no Mac OS version)

The Parallax Side-Scroller is a classic shoot'em up game with a virtual joypad and button provided by SneakyInput. The project shows you how to create an endlessly scrolling parallax background, how to structure bigger projects into multiple classes, how to pool and re-use game objects, how to keep nodes within a defined area by overriding setPosition, how to create reusable game components, how to turn your game's scene into a Singleton to make it accessible to other classes, and how to access other classes through your game's scene.

Chapters 6-8

Pinball Game

iOS



Mac OS

The Pinball project is a functional pinball table with bumpers, flippers, a plunger and a ball. Box2D is used for physics simulation, the collision shapes were created with PhysicsEditor. The plunger and flippers are animated via Box2D motors. Collision events are forwarded via Objective-C messages to individual objects and correspondingly named selectors (eg beginContactWithBall, endContactWithPlunger). Includes a demonstration of gravitational pull by uncommenting the function applyForceTowardsFinger.

Chapter 13

Creating a Kobold2D Template Project

You can create your own template projects with ease. Template projects are projects that you use as basis for new projects, with more or less boilerplate code already included. Template projects will be listed in the Kobold2D Project Starter tool as described in Creating a Kobold2D Project.

Reasons for creating your own Template Projects?
You may want to share your own template projects with the community, maybe with an accompanying tutorial. This is a sure way to get attention.

Businesses can use template projects to ensure that new projects include all the shared source code, assets, build schemes, etc. that you have developed over time, or need to have in every project (eg. OpenFeint).

You can also use template projects as starting points for the customers of your commercial add-on library for Kobold2D.

Before you begin ...

It is highly recommended to build a template project based on one of the already existing template projects provided by Kobold2D. After all, you want to end up with a working Kobold2D project.

If in doubt ...
... check how the existing Kobold2D project templates are setup. You can open the template projects in the `/__Kobold2D__/templates/project/` folder and review the project, but it won't build. To be sure you don't accidentally break the template project, copy it elsewhere before opening the .xcodeproj file.

Correctly Naming a Template Project

Template projects rely on a common naming scheme for the project's folder, Xcode project, targets and schemes. The naming scheme is as

follows:

_ + The-Project-Name + -Template_

For example if your template project's name is supposed to be "Hello My Friend" then the project name will be `_Hello-My-Friend-Template_` . The rest of this guide assumes that `_Hello-My-Friend-Template_` will be the chosen name for the template, but of course you can change the `Hello-My-Friend` part to whatever you like.

It is recommended to replace all spaces with dashes, and to avoid using any punctuation or other special characters. Template Projects and the Kobold2D Project Starter tool have only been tested with names containing underscores, dashes, letters and digits.

When you create a new project by using the Kobold2D Project Starter tool, all occurances of the string `_Hello-My-Friend-Template_` will be replaced by the project name the user chooses.

This replacement is done for the project's folder, the .xcodeproj file, the project's targets, schemes and groups. This replacement however does **not** extend to source code or resource files.

To summarize, make sure that the following is true in your template project:

- the project and its files are in a folder named: `_Hello-My-Friend-Template_`
- the project file (.xcodeproj) is in that folder and named: `_Hello-My-Friend-Template_.xcodeproj`
- the project's targets are named: `_Hello-My-Friend-Template_-iOS` for iOS targets and `_Hello-My-Friend-Template_-Mac` for Mac targets. The `-iOS` and `-Mac` suffixes are optional but help to identify the target platform.
- the project's schemes are named exactly like the project's targets: `_Hello-My-Friend-Template_-iOS` and `_Hello-My-Friend-Template_-Mac`

If your template project only supports one platform, simply omit the target and scheme for the platform(s) that the project doesn't support.

Adding a Description

To add a description that will be displayed by the Kobold2D Project Starter tool, simply add a text file named `_description_` in the same folder as the .xcodeproj file. Verify that the description text fits into the description box of the Kobold2D Project Starter tool.

Moving the Template Project

After you've finished renaming your template project, along with all other work, you will have to move or copy the project's folder to the project templates folder `/__Kobold2D__/templates/project/`.

Now you can run the Kobold2D Project Starter app. Your template project should show up. Select it and create a new project based on your template project, preferably give it a name that stands out (eg. `TEMPLATETEST`) because that will make the next step easier.

Verify that the project folder, Xcode project, targets and schemes have been properly renamed to `TEMPLATETEST` or whatever name you chose. The targets and schemes should be named `TEMPLATETEST-iOS` and `TEMPLATETEST-Mac`.

Building and Running the Template Project

If the names have all been correct replaced, verify that the `TEMPLATETEST` project builds without errors.

If the `TEMPLATETEST` project builds successful it will be sufficient to test future changes to your template project in the template project *`Hello-My-Friend-Template`* itself. You only occasionally will have to verify that the project also builds without errors after going through the Kobold2D Project Starter tool.

Open the **Product** menu in Xcode, hold down the Option key and select **Clean Build Folder ...** to make sure everything gets build anew. You only have to do this step once before building any of your newly created project's targets.

You will want to build each target (iOS & Mac) once, and run it to see that it behaves as it should. You should build and run the iOS target once for each of the possible selections: iOS Device, iPhone Simulator and iPad Simulator. Even if your template does not support iPad its good practice to make sure it at least runs on iPad.

You also will want to verify that the project behaves correctly on both standard and Retina displays. Run the project once using the iPhone Simulator as target. Verify the app is correct, then in the Simulator go to the **Hardware -> Device** menu and switch to the other iPhone type, which will be either **iPhone** or **iPhone (Retina)**. Then hit the run button in Xcode again, or simply tap the app's icon in the Simulator. This way you can quickly ensure that your app displays correctly on standard and Retina devices.

Finally, you should build and run the Mac targets for both 64-Bit and 32-Bit modes, assuming your installation of Mac OS X supports both. This is important because sometimes compile or runtime errors will only occur in either 32-bit or 64-bit builds.

One more reason to share template projects...
... the best, most popular template projects created by the community will be included in Kobold2D if the author(s) allow it to be distributed under the MIT License.

You're done!

You can now share the template project with your team or the entire community. Right-Click the project's `_Hello-My-Friend-Template_` folder and select *Compress "_Hello-My-Friend-Template_}}. You can rename the archive to whatever you like.

Provide your users with installation help by linking to this article:
Installing a Kobold2D Template Project

Installing a Kobold2D Template Project

If you receive an archive containing a Template Project for Kobold2D, this quick guide explains how to install it.

Developers of template projects should include the following link along with the template download link (on your webpage, forum post, or readme file) as instruction for users of their template project:
Installing a Kobold2D Template Project — `http://www.kobold2d.com/x/ZwUO`

After downloading and extracting an archive containing a Kobold2D template project you should notice a new folder named similar to this example: `_Hello-My-Friend-Template_`

Copy or move the folder to the Kobold2D templates folder. By default the templates folder is located here:

```
~/Kobold2D/Kobold2D-x.y/__Kobold2D__/templates/project/
```

Whereas Kobold2D-x.y will be the versioned Kobold2D folder, for example: `Kobold2D-1.0`

If the project template was created for an older version of Kobold2D and hasn't been maintained, it's probable that projects based on this template project may not compile, contain bugs, or won't even show up in the Kobold2D Project Starter tool.

After copying the folder you can launch the Kobold2D Project Starter tool and start a new project based on the newly installed project template.

If the template project does not show up, and in particular the template's folder does not begin with *{} (underscore) and end in `-Template`* please contact the project template author for support.

Xcode Build Locations Setup for Kobold2D

Are you wondering why this page opened?
This page may be opened *automatically during a Kobold2D build*. Kobold2D does so because it determined that your Xcode Build Location setting is set to `Locations Specified by Targets` (in Xcode 4.3 called `Legacy`). Please read the entire page because it contains crucial information for you to build Kobold2D successfully.

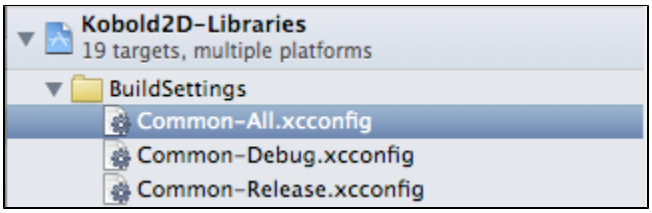
Kobold2D does not build out of the box if your Xcode **Build Location** setting is set to `Locations Specified by Targets` (Xcode 4.3: `Legacy`) as indicated by this Xcode Build Error:

This article describes the quick fix and the recommended solution. It also explains the reasons why the recommended solution isn't recommended for nothing, and why the `Locations Specified by Targets` (Xcode 4.3: `Legacy`) setting **should only be used** if you are — for whatever reason — required to use it.

- The Quick Fix (read the caveats!)
- The Recommended Fix
  - Why is "Derived Data Location" / "Unique" recommended?
  - I don't like the convoluted DerivedData path …
  - I must use the "Locations Specified by Targets" setting …

The Quick Fix (read the caveats!)

open the `Common-All.xcconfig` file:



remove the // comment from the `//SYMROOT = ..` line:

```
SYMROOT = ~/Kobold2D/build
```

You do not have to use the default `~/Kobold2D/build` location, you can use any other folder location. However you should not use the location or a subfolder of a Kobold2D installation or any Xcode project.

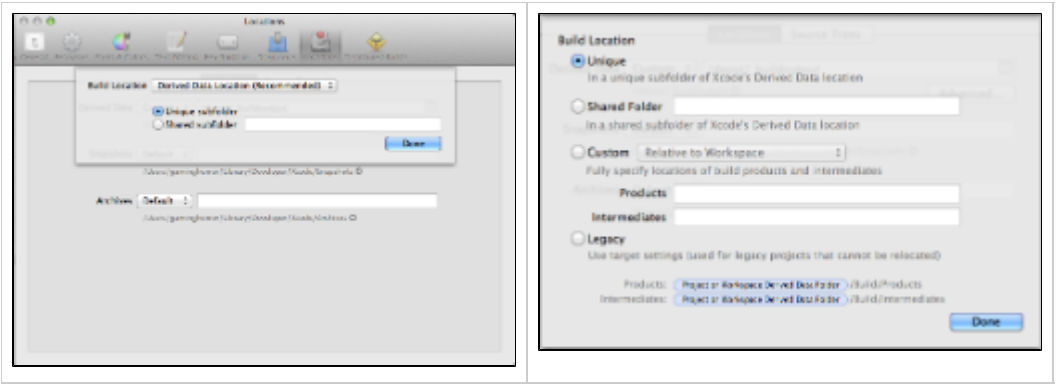
Quick Fix Caveats
You must issue a **Product -> Clean** every time you switch to a project using a different Kobold2D version than the version that was most recently built. Otherwise you may experience linker errors (undefined symbol) or worse, inexplicable crashes. This is why the **quick fix is not recommended for continued use**.

The Recommended Fix

Unless you are explicitly required (as per instructions of a specific library, or as per requirement from your employer) simply do **not** use the `Locations Specified by Targets` (Xcode 4.3: `Legacy`) setting.

Instead, change it back to the recommended setting `Derived Data Location (Recommended)` (Xcode 4.3: `Unique` or any other setting other than `Legacy`). You can make this change in **Xcode -> Preferences -> Locations -> Advanced**.

Xcode 4.0 through 4.2	Xcode 4.3 and newer



Why is "Derived Data Location" / "Unique" recommended?

Xcode 4 introduced the "Derived Data" location to solve long-standing issues of stale code and cleaning builds affecting other projects.

By using the default Build Location setting `Derived Data Location (Recommended)` (Xcode 4.3: `Unique`) Xcode 4 avoids prevalent issues present in Xcode 3, specifically when linking with static libraries:

Xcode is guaranteed to link with the correct static library, as each library is placed in a derived data folder unique to each project.

In Xcode 3 respectively Xcode 4 with `Locations Specified by Targets` (Xcode 4.3: `Legacy`) the library is placed in a common folder which may not be updated when the library changes, or when another project uses a different version of the library. This can lead to linker errors (undefined symbol) or worse, it can cause inexplicable runtime crashes. This problem is almost certainly going to surface once you start working with multiple versions of Kobold2D (ie after upgrading).

This is the main reason why `Derived Data Location (Recommended)` (Xcode 4.3: `Unique`) is the recommended setting not just for Kobold2D but for any Xcode project.

When issuing a Product -> Clean, only the current build configuration for the current project is cleaned.

In Xcode 3 respectively Xcode 4 with `Locations Specified by Targets` (Xcode 4.3: `Legacy`), all build configurations are cleaned and all projects linking to the same libraries are affected by the clean as well. In effect more code than necessary will have to be compiled again. This is simply a waste of your time.

> Xcode 4 can still clean all of your projects. By selecting **Product -> Clean** while holding down the option key the menu item changes to **Clean Build Folder…** which will delete all your project's build output files.

Separating the "build" output folder from the project folder means it can not be accidentally added to source control.

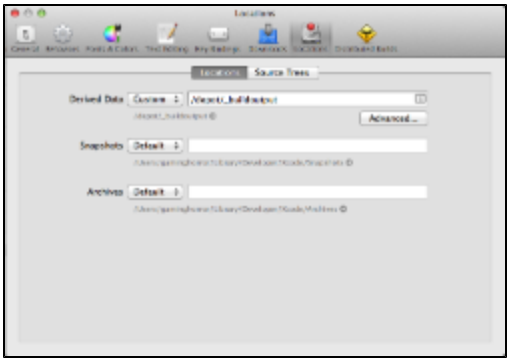
Having build output files in source control can have dire consequences, such as: wasted space, bloated change history, permission issues, build files out of synch after source control operations, utterly confused Xcode.

> **Unless instructed otherwise, use "Derived Data Location (Recommended)" / "Unique"**
> The above issues should explain why Xcode itself labels the Derived Data / Unique option as **(Recommended)**. You should use the recommended setting unless you are absolutely, irrevocably required to use `Locations Specified by Targets` (Xcode 4.3: `Legacy`).

I don't like the convoluted DerivedData path …

The majority of developers using `Locations Specified by Targets` (Xcode 4.3: `Legacy`) do so merely because they dislike the convoluted path to the DerivedData folder and/or because that's where Xcode 3 used to place build output files. This should **never** be a good enough reason to use this setting.

1) You can use the **Custom** or **Relative** setting for the Derived Data location to make the folder easier to find without negatively affecting builds.



2) You can quickly open the Derived Data folder through **Window -> Organizer** by selecting the desired project, then click the arrow symbol to the right of the Derived Data path to open the folder in a Finder window.

I must use the "Locations Specified by Targets" setting …

If there's a specific reason to use `Locations Specified by Targets` (Xcode 4.3: `Legacy`), usually a requirement of a (outdated?) library or a (uninformed?) employer, then you can only do one of these things:

1) Change the setting back and forth depending on the project you're working on. You can change the corresponding setting `IDEBuildProductsLocationDeterminedByTargets` directly in `~/Library/Preferences/com.apple.dt.Xcode.plist`.

The setting can be changed while Xcode is running, so you may want to consider writing a script. You may want to try adding this script as a Run Script build phase to each project, it may or may not take effect for the current build but it will be in effect if you build a second time.

2) Bite the bullet, uncomment the SYMROOT line as instructed above and keep the caveats in mind. If you get linker errors or inexplicable crashes **after switching, renaming or relocating projects**, get into the habit of cleaning the build and try again before attempting anything else.

3) If you don't need the source code of the library that requires the `Locations Specified by Targets` (Xcode 4.3: `Legacy`) setting, you can build it once and then just link with the binary file that you placed to a common location on your hard drive.

4) Assuming the library in question is still being maintained, petition to update the library so that it supports proper Xcode 4 build locations. Or, if possible, spend some time to come up with the fix yourself.

5) Ask your employer what the rationale is for using `Locations Specified by Targets` (Xcode 4.3: `Legacy`). This may simply be an old habit, and as you may know, those die hardest. Maybe you can convince your boss that this setting is counterproductive and a potential waste of precious developer time.

Kobold2D Programming Guide

The Kobold2D Programming Guide contains instructions on how to use the exclusive Kobold2D features through how-to guides, example code, and direct links to relevant classes in the API References.

- Config.lua Settings Reference — The config.lua resource file is used to control the startup configuration of your project. You can also use it to store arbitrary game data, for example to make it available to non-programming team members.
- Working With Ad Banners — Kobold2D has built-in support for Ad Banners from either iAd or AdMob. This article explains how to enable and use Ads in your app.
- Processing User Input — Kobold2D provides a simple-to-use, one-stop class for handling all your user input processing needs: **KKInput** (class reference). The User-Input template projects illustrates the use of KKInput for processing user input.
- Working with Automatic Reference Counting (ARC) — This guide will introduce you to automatic reference counting (ARC). ARC is fully supported by Kobold2D and enabled by default in all template projects. If you start a new Kobold2D project, you're already using ARC. Of course you can disable ARC but you shouldn't really need to.

Config.lua Settings Reference

The config.lua resource file is used to control the startup configuration of your project. You can also use it to store arbitrary game data, for

example to make it available to non-programming team members.

The KKStartupSettings in the **config.lua** file included in each Kobold2D project are described in detail in the Kobold2D API Reference:
KKStartupSettings class reference

Note that the KKStartupSettings properties begin with a lowercase character which must be written as uppercase in the config.lua file:

KKStartupSettings Property name:	config.lua Parameter name:
gLViewColorFormat	GLViewColorFormat
deviceOrientation	DeviceOrientation
enableMultiTouch	EnableMultiTouch

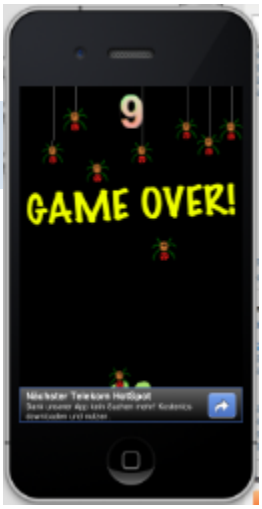
KKStartupConfig Reference

KKStartupConfig Reference for Kobold2D 2.0
KKStartupConfig Reference for Kobold2D 1.x

Could not retrieve RSS feed: no URL

Working With Ad Banners

Configurable Ad Banner support is currently disabled in Kobold2D 2.0. The KKAdBanner class still exists, but it's no longer used as a built-in feature.



Kobold2D has built-in support for Ad Banners from either iAd or AdMob. This article explains how to enable and use Ads in your app.

Interstitials are currently not supported.

When enabled, an Ad Banner view will slide in either from the top or bottom as soon as the Ad network has provided the first Ad. The Ad Banner view will be hidden if the user rotates the device, or an Ad fails to load. For iAd, hiding ad banners without content is a requirement for approval and correctly handled by Kobold2D.

- Configuring Ad Banners
- Setting up Apple iAd
  - iAd Resources
- Setting up AdMob
  - AdMob Resources

Configuring Ad Banners

Ad Banners can be enabled and configured in your project's config.lua file:

```
EnableAdBanner = NO,
PlaceBannerOnBottom = YES,
LoadOnlyPortraitBanners = NO,
LoadOnlyLandscapeBanners = NO,
AdProviders = "iAd, AdMob",
AdMobRefreshRate = 15,
AdMobPublisherID = "YOUR_ADMOB_PUBLISHER_ID",
AdMobTestMode = YES,
```

If `EnableAdBanner` is YES, your app will start displaying Ad banners.

With AdProviders you specify the Ad providers you want to use and their priority. The default is:
`AdProviders = "iAd, AdMob",`

This means that iAd will be used where available (devices running iOS 4.0 or newer), while AdMob is only used where iAd isn't available (iOS 3.x). If you want to use iAd or AdMob exclusively, simply change the setting to "iAd" or "AdMob". The setting "AdMob, iAd" is meaningless because AdMob will work on all devices, it will never fall back to iAd.

> If the selected ad network is temporarily unavailable no ads will display. There is no "ad provider network outage fallback solution" in this implementation. If you find that important, please submit a request.

The Ad Provider specific API is wrapped in a common class KKAdBanner (class reference). KKAdBanner allows you to load or unload banners (show and hide), and it gives you access to the respective Ad Provider's banner view: iAdBannerView respectively adMobBannerView. Either property can be nil, depending on which is enabled in the config.lua and which is available on the device.

Setting up Apple iAd

For iAd to work you must first join the iAd Network, then setup your app to support iAd. Both can be done through iTunes Connect. See the iTunes Connect Developer Guide for detailed instructions.

> Getting payments for iAd ads requires you to fill out Contracts, Tax, and Banking information online in iTunes Connect. The iAd contracts are separate from the Paid Applications contract.

`LoadOnlyPortraitBanners` and `LoadOnlyLandscapeBanners` can be used to instruct the iAd banner view to only load banners for either device orientation. At least one of these options must be set to YES. Use these settings if your app only supports one type of orientation (landscape or portrait) to conserve network bandwidth and to allow ads to load faster.

iAd Resources

iAd Network (App Store Resource Center)
iAd Programming Guide
iAd Framework Reference
iTunes Connect Developer Guide

Setting up AdMob

AdMob is available on all devices running iOS 3.0 or newer. It is easier to setup and allows you to get paid via PayPal. No contracts or tax information need to be filled out. Because of that and developers reporting better revenue through AdMob than iAd, it seems to be more widely used than iAd.

> Thanks to Simon Jewell for providing the initial AdMob Kobol2D implementation.

For AdMob to work you need to setup a Publisher ID which is also called "Ad Unit ID". You will see the latter referenced in code respectively log files, for example if you're using an incorrect Publisher ID the warning will state that the "Ad Unit ID" is wrong.

Setup your AdMob Publisher account with the instructions provided in this link. Setup an App and paste the app's Publisher ID into the corresponding setting (within the brackets) so it looks something like this:

```
AdMobPublisherID = "af234e8bcfc2",
```

You can also configure the refresh rate, ie how long an Ad is displayed before the next Ad is loaded. AdMob allows you to set a range of 12 to 120 seconds, Kobold2D default is 15:

```
AdMobRefreshRate = 30,
```

You can configure the time it takes before the very first AdMob Ad is loaded. With that you have some control over when the first Ad appears, for example to decrease the likelihood that the first Ad is shown during the App's loading screen.

```
AdMobFirstAdDelay = 5,
```

You can also specify that Test Mode should be enabled (default):

```
AdMobTestMode = YES,
```

Test Mode prevents AdMob from generating false impressions. You should leave this setting on at all times, except for when you want to see real ads while debugging. You do not need to set AdMobTestMode to NO when building your app for release (archive) as the setting will automatically be ignored (assumed to be NO) in release builds.

The size of an app (archive build) with AdMob grows by around 900 KB. Since not everyone wants to use AdMob, and some users would rather have a smaller app, there's a preprocessor macro to prevent KKAdBanner from including the AdMob code and library.

In the **Kobold2D Libraries** project, expand the Kobold2D group, and open the file *kkUserConfig.h*. You need to comment out (or delete) the following line to disable AdMob and reduce your App size:

```
// comment this line to remove the Kobold2D AdMob code
#define KK_ADMOB_SUPPORT_ENABLED
```

This change affects all projects using that particular Kobold2D version.

AdMob Resources

How to setup your AdMob Publisher account
Google AdMob Ads iOS Fundamentals (Kobold2D already covered this for you)
Google AdMob Ads iOS Intermediate (again, already covered)

Processing User Input

Kobold2D provides a simple-to-use, one-stop class for handling all your user input processing needs: **KKInput** (class reference). The User-Input template projects illustrates the use of KKInput for processing user input.

What's special about KKInput?

KKInput allows you to access the user input devices / states in any class and method at any time during the game loop.

Normally input is processed by listening to events (callback methods), and most of the input state is only available during that event, unless you store the event data somehow. KKInput does that for you, it remembers the input states and allows you to access them when and where you need them through a clean and simple interface.

KKInput can be used without having to learn and understand the iOS or Mac OS SDK features that power KKInput. Commonly used features like accelerometer filtering and properly converting touch and mouse locations to the Cocos2D OpenGL view are performed behind the scenes, so you don't have to.

In addition KKInput doesn't care which platform you're currently targeting. You do not have to #ifdef platform-specific input code. For example, calling a keyboard or mouse method in an iOS build will compile without issues and the keyboard/mouse methods will return safe default values if they happen to be used on an iOS device.

KKInput Documentation

- KKInput General Information
- iOS:
  - Touch and Multi-Touches
  - Gestures (Tap, Double-Tap, Long-Press, Swipe, Pan, Rotation, Pinch)
  - Accelerometer with High- & Low-Pass Filter
  - Gyroscope
  - Device Motion (accelerometer & gyroscope sensor fusion)
- Mac OS:
  - Keyboard
  - Mouse with Scroll Wheel

KKInput General Information

- Platform-Specific User Input
- Singleton access
- Enable/Disable User Interaction for iOS
- "ThisFrame" properties and methods
- KKInput Performance Notes

Platform-Specific User Input

You should not use #ifdef to encapsulate platform specific input with KKInput to ensure that as much of your codebase as possible is compiled for

both platforms, to avoid dreadful compiler errors when you switch to building for the other platform.

If you need to process input separately based on the current platform, use these Kobold2D CCDirector extensions:

```
CCDirector* director = [CCDirector sharedDirector];
if (director.currentPlatformIsIOS)
{
    // process iOS user input
}
else if (director.currentPlatformIsMac)
{
    // process Mac OS user input
}
```

Singleton access

KKInput is a Singleton, you can get its instance via `[KKInput sharedInput]`. Often times you'll need to access KKInput methods repeatedly, in that case you would want to keep the reference local to reduce the number of sharedInput messages (method calls), like so:

```
KKInput* input = [KKInput sharedInput];
if (input.touchesAvailable)
{
    // ... more input processing follows
}
```

Enable/Disable User Interaction for iOS

To enable or disable user input processing for the Cocos2D OpenGL view, you can either set `EnableUserInteraction = YES` in config.lua or set the `userInteractionEnabled` property:

```
[KKInput sharedInput].userInteractionEnabled = YES;
```

The property has no effect when building a Mac app.

"ThisFrame" properties and methods

Various properties/methods carry the "ThisFrame" suffix in their name to indicate that this is an event which is only true for the duration of the current frame. For example, if a touch ends, it is a discrete event that is true in this frame and false again in the next and all following frames.

The significance of this is that if you use a "ThisFrame" property or method in a method that runs only once, for example the init method of a class, then the "ThisFrame" method will not be true in all likelihood. For it to be true the user would have to, for example touch the screen at the exact same frame as your class runs the init method.

Instead, only check for the occurrence of "ThisFrame" events only in scheduled selectors, and in particular in scheduled selectors without specifying an interval. The alternative would be to check for the continuous event instead.

KKInput Performance Notes

KKInput does a lot behind the scenes to keep the performance fast.

All objects that are often needed are pooled, ie they remain in memory at all times to avoid frequent alloc/dealloc cycles. This relates to touches, key and mouse states in particular. KKInput allocates a number of these objects in advance, ie five touches and a buffer of up to 16 simultaneous key presses (this is more than most keyboards can handle correctly by the way) and simply re-uses them as needed.

The necessary conversion of locations to Cocos2D view coordinates and the filtering of acceleration values for example is only done the moment you access these values, and only once per frame. This prevents the calculations from being performed unnecessarily multiple times per frame if you happen to access the values more than once per frame.

KKInput Touches

- Enable Multi-Touch
- Simple Touch Began/Ended Tests
- Touches Available
- Did you just touch my Node???
- Location of Touches
- Accessing all Touches

- Removing a Touch

Enable Multi-Touch

To enable multiple touches, either set the config.lua setting `EnableMultiTouch = YES` or set the `multipleTouchEnabled` property:

```
[KKInput sharedInput].multipleTouchEnabled = YES;
```

Simple Touch Began/Ended Tests

You can quickly test if any touch began or ended in the current frame with the following methods. This is a quick & easy way to, for example, create screens which the user can dismiss by touching the screen anywhere, or having the user "touch the screen to play again".

```
if ([KKInput sharedInput].anyTouchBeganThisFrame) { ... }
if ([KKInput sharedInput].anyTouchEndedThisFrame) { ... }
```

Touches Available

To test if touches are available, ie if any finger is touching the screen, use `touchesAvailable`:

```
if ([KKInput sharedInput].touchesAvailable) { ... }
```

Did you just touch my Node???

You can determine if a node is being touched (oh my) with the `isAnyTouchOnNode:touchPhase:` method. You pass in the node like a CCSprite, CCLabelTTF, or any other CCNode derived object to test if any touch with the given phase (began, ended, any, etc) was on that node.

The `isAnyTouchOnNode` method correctly performs the test even if the node is rotated, scaled or both.

```
KKInput* input = [KKInput sharedInput];
if ([input isAnyTouchOnNode:someNode touchPhase:KKTouchPhaseBegan]) {...}
if ([input isAnyTouchOnNode:someNode touchPhase:KKTouchPhaseAny]) {...}
```

Instances of CCNode by definition have a zero contentSize, so you can not use this method to test for touches on CCNode, unless you have modified the contentSize property of the CCNode object.

CCScene or CCLayer by definition have a contentSize that equals that of the screen/window size. You can test if a touch was on a CCScene or CCLayer object, but it makes little sense because if there is a touch on the screen, it will naturally be on the CCScene or CCLayer object — unless you have modified the contentSize property.

Location of Touches

You can get the location of any touch with `locationOfAnyTouchInPhase:`. You can use one of the standard phases (began, moved, stationary, ended, cancelled) or use the special Kobold2D phase `KKTouchPhaseAny` to test for the location of any touch in any phase.

When is this useful? Whenever you don't care which finger is at that location, or if multiple touches are disabled you can get the location of the one (any) finger on the screen.

```
KKInput* input = [KKInput sharedInput];
CGPoint pos = [input locationOfAnyTouchInPhase:KKTouchPhaseAny];
```

If no finger is currently touching the screen that is in the given phase, then CGPointZero is returned.

Accessing all Touches

The most powerful way to work with touches is to obtain the list of currently available touches via the `touches` property, which is a CCArray* containing zero to five KKTouch* objects.

KKTouch is a wrapper for UITouch that allows you to work with touches already converted to the Cocos2D view space, and without having to use #ifdef if you also build for Mac. You can get the KKTouch* array and iterate over it like this:

```
CCArray* touches = [KKInput sharedInput].touches;
KKTouch* touch;
CCARRAY_FOREACH(touches, touch)
{
    // do something with touch here
    mySprite.position = touch.location;
}
```

The KKTouch `location` and `previousLocation` properties are already converted to Cocos2D view coordinates. You do not need to (and shouldn't) call CCDirector's convertGL method.

Each KKTouch object corresponds to a particular finger on the screen. The same KKTouch object is used for the same finger starting when the touch "began" until the touch either "ended" or got "cancelled".

You can not rely on the array index for accessing a particular finger's KKTouch as other touches may end and get started. You can either store and later compare the `touchID` property to make sure you're using the same KKTouch as before, or simply keep a weak reference to the KKTouch object. In that case you have to check that it's `phase` property isn't `KKTouchPhaseLifted`, which indicates that the KKTouch object is no longer associated with a finger on the screen. You will also want to nil your reference when the touch phase is either `KKTouchPhaseEnded` or `KKTouchPhaseLifted`.

If you ever need access to the underlying UITouch* object, you can simply cast the `touchID` property:

```
// access the underlying UITouch object if needed
UITouch* uiTouch = (UITouch*)touch.touchID;
```

Do not retain or otherwise hold on to UITouch* objects, they may contain invalid data in the next frame. Access them only through the `touches` property on a "as needed" basis.

Removing a Touch

As of Kobold2D v1.0.2 you can remove a touch to prevent other classes from handling the same touch. The finger associated with that KKTouch object will be ignored from then on.

```
CCArray* touches = [KKInput sharedInput].touches;
KKTouch* touch;
CCARRAY_FOREACH(touches, touch)
{
    // remove touch
    [[KKInput sharedInput] removeTouch:touch];
}
```

It is legal to remove touches while iterating over the touches array. The touch is first invalidated (KKTouchPhase is set to `KKTouchPhaseLifted` among other things) and later removed.

KKInput Gestures

- KKInput Gesture Recognizer Support
- Gestures can influence each other
- Testing if Gesture Recognizers are Available
- Enabling Gestures
- Gesture Location
- Discrete Gestures: Tap, Double-Tap and Swipe
- Continuous Gestures: LongPress, Pan, Rotation and Pinch
- Swipe Gesture Properties
- Pan Gesture Properties
- Rotation Gesture Properties
- Pinch Gesture Properties
- Cancelling a Continuous Gesture Recognizer
- Accessing the underlying UIGestureRecognizer objects

KKInput Gesture Recognizer Support

KKInput supports all built-in UIGestureRecognizer gestures:

Gesture	Tap	Double-Tap	Long-Press	Swipe	Pan	Rotation	Pinch
Discrete	✓	✓	✗	✓	✗	✗	✗
Fingers	1	1	1	1	1	2	2

The Tap, Double-Tap and Swipe gestures are discrete events that are true only during the frame the gesture was recognized. All other gestures are continuous, meaning they continue to update their state until the gesture ends.

> ⚠ KKInput only supports single-finger gestures, with the exception of Rotation and Pinch which require two fingers by definition. While you can create recognizers for two, three or more finger taps, long-presses, swipes and so on, using such gestures in games has very limited application. If you have a good use case for such gestures, feel free to make a request in the Kobold2D Feedback section.

Gestures can influence each other

Not all gestures can be enabled at the same time without influencing each other.

- **Tap and Double-Tap**
  - If both are enabled, the Double-Tap gesture must fail before the Tap gesture is recognized. This introduces an unavoidable delay before the Tap gesture is recognized.
- **Pan and Swipe**
  - If both are enabled, they will be recognized simultaneously.
- **Rotation and Pinch**
  - If both are enabled, they will be recognized simultaneously. This means that you can implement a rotate & zoom action by enabling both gestures at the same time.
  - ⚠ You should not use Pan or Long-Press simultaneously with Rotation and Pinch, because doing so will make it difficult for the user to correctly initiate the rotation or pinch gesture.

Testing if Gesture Recognizers are Available

Gesture Recognizers and thus KKInput gestures are available on iOS 3.2 and newer. To test if the current device supports gestures check the `gesturesAvailable` property:

```
if ([KKInput sharedInput].gesturesAvailable) {...}
```

Enabling Gestures

Each gesture can be enabled/disabled separately. You will have to set the appropriate "Enabled" properties. If gestures are not supported on the current device (pre iOS 3.2), gestures will not be enabled.

```
KKInput* input = [KKInput sharedInput];
input.gestureTapEnabled = YES;
input.gestureDoubleTapEnabled = YES;
input.gestureLongPressEnabled = YES;
input.gestureSwipeEnabled = YES;
input.gesturePanEnabled = YES;
input.gestureRotationEnabled = YES;
input.gesturePinchEnabled = YES;
```

Gesture Location

All gestures have their own `gestureXXXLocation` property which returns the location of the gesture in Cocos2D OpenGL view coordinates.

For Tap and Double-Tap it is the location where the finger touched the screen. For Swipe it is the location where the Swipe gesture started.

For LongPress and Pan gestures it is the current location of the finger. For Rotation and Pinch the location is the middle point between the two fingers.

```
CGPoint pos;
KKInput* input = [KKInput sharedInput];
pos = input.gestureTapLocation;
pos = input.gestureDoubleTapLocation;
pos = input.gestureLongPressLocation;
pos = input.gestureSwipeLocation;
pos = input.gesturePanLocation;
pos = input.gestureRotationLocation;
pos = input.gesturePinchLocation;
```

Discrete Gestures: Tap, Double-Tap and Swipe

The Tap, Double-Tap and Swipe gestures are discrete, ie they're only true in the frame they happened. Therefore they can only be checked with "ThisFrame" properties:

```
KKInput* input = [KKInput sharedInput];
if (input.gestureTapRecognizedThisFrame) {...}
if (input.gestureDoubleTapRecognizedThisFrame) {...}
if (input.gestureSwipeTapRecognizedThisFrame) {...}
```

Only when a "ThisFrame" property is true will the corresponding `gestureXXXLocation` property contain a meaningful location.

Continuous Gestures: LongPress, Pan, Rotation and Pinch

All continuous gestures have a `gestureXXXBegan` property which is true the moment the gesture was first recognized, and stays true until the gesture ends or is cancelled.

```
KKInput* input = [KKInput sharedInput];
if (input.gestureLongPressBegan) {...}
if (input.gesturePanBegan) {...}
if (input.gestureRotationBegan) {...}
if (input.gesturePinchBegan) {...}
```

Only while the "Began" property is true will the corresponding `gestureXXXLocation` property contain a meaningful location.

Swipe Gesture Properties

The swipe gesture has an additional property `gestureSwipeDirection` of type `KKSwipeGestureDirection` with which you can determine the direction of the swipe: right, left, up or down.

```
KKInput* input = [KKInput sharedInput];
KKSwipeGestureDirection dir = input.gestureSwipeDirection;
switch (dir)
{
    case KKSwipeGestureDirectionRight:
        // direction-specific code here
        break;
    case KKSwipeGestureDirectionLeft:
        // direction-specific code here
        break;
    case KKSwipeGestureDirectionUp:
        // direction-specific code here
        break;
    case KKSwipeGestureDirectionDown:
        // direction-specific code here
        break;
}
```

Pan Gesture Properties

The Pan gesture has two extra properties `gesturePanTranslation` and `gesturePanVelocity`. The translation is a CGPoint that contains how far, in points, the finger has moved relative to the point where the Pan gesture began. The velocity is a CGPoint that tells you how fast the finger is currently moving across the screen, in points per frame.

```
KKInput* input = [KKInput sharedInput];
CGPoint translation = input.gesturePanTranslation;
CGPoint velocity = input.gesturePanVelocity;

input.gesturePanTranslation = newTranslation;
```

You can also modify the `gesturePanTranslation` property. If you do so, the `gesturePanVelocity` property is reset.

Rotation Gesture Properties

The Rotation gesture knows two exclusive properties `gestureRotationAngle` and `gestureRotationVelocity`. The angle is is the current rotation angle, in degrees (0-360). You can directly assign the angle to a node's `direction` property. The velocity tells you how fast the user is performing the rotation gesture, in degrees per frame.

```
KKInput* input = [KKInput sharedInput];
aSprite.direction = input.gestureRotationAngle;
float velocity = input.gestureRotationVelocity;

input.gestureRotationAngle = 180.0f;
```

You can also modify the `gestureRotationAngle` property. If you do so, the `gestureRotationVelocity` property is reset.

Pinch Gesture Properties

The Pinch gesture has two properties `gesturePinchScale` and `gesturePinchVelocity`. Scale is the relative scale factor of the two fingers, it increases the further the two fingers move apart, it decreases the closer they move towards each other. The scale property can be directly assigned to a node's `scale` property. The velocity is how fast the fingers are moving towards or away from each other, in scale factor per frame.

```
KKInput* input = [KKInput sharedInput];
aSprite.scale = input.gesturePinchScale;
float velocity = input.gesturePinchVelocity;

input.gesturePinchScale = 2.0f;
```

You can also modify the `input.gesturePinchScale` property. If you do so, the `input.gesturePinchVelocity` property is reset.

Cancelling a Continuous Gesture Recognizer

If you want to ignore a continuous gesture after it began, you can cancel it simply by disabling and re-enabling the gesture like this:

```
if (input.gesturePanBegan)
{
   // don't like continuing with the gesture? Reset it:
   input.gesturePanEnabled = NO;
   input.gesturePanEnabled = YES;
}
```

Note that this will release the previous gesture recognizer and allocate a new one, so you should avoid doing that every frame.

Accessing the underlying UIGestureRecognizer objects

As of Kobold2D v1.0.2 KKInput has the following properties with which you can gain access to the UIGestureRecognizer instances used by KKInput. This allows you to use all of the gesture recognizer features supported by UIGestureRecognizer and concrete subclasses, since KKInput only exposes the most frequently needed features.

```
UITapGestureRecognizer *  tapGestureRecognizer;
UITapGestureRecognizer *  doubleTapGestureRecognizer;
UILongPressGestureRecognizer *  longPressGestureRecognizer;
UIPanGestureRecognizer *  panGestureRecognizer;
UIRotationGestureRecognizer *  rotationGestureRecognizer;
UIPinchGestureRecognizer *  pinchGestureRecognizer;
```

Note that these properties may return nil if the gesture recognizer is not enabled.

You should not retain these properties! The gesture recognizers are retained by KKInput until you disable the gesture. Once you re-enable the gesture, a new gesture recognizer is allocated which would then be used rather than your retained instance.

In Mac OS X builds these properties are declared to be of type `id` and will always return nil.

By default the delegate property of the gesture recognizers is the KKInputGesture class. Therefore you should not modify the delegate or KKInput would stop processing that gesture. If you need to use your own delegate you'll be better off to just create the instance of UIGestureRecognizer yourself and not use KKInput's gestures. You may want to use some of the code in the handleXXXGesture methods in the KKInputGesture class though.

KKInput Accelerometer

KKInput supports accelerometer input on all devices, using either UIAcceleration or Core Motion.

- Enabling Accelerometer
- Accessing Acceleration Values
- About Filtering

Enabling Accelerometer

To enable or disable accelerometer input use the `accelerometerActive` property. You should disable accelerometer input whenever you don't need it to conserve battery power of the user's device.

```
KKInput* input = [KKInput sharedInput];
input.accelerometerActive = YES;

if (input.accelerometerAvailable) {...};
```

You can also check for accelerometer availability with `accelerometerAvailable`. While this may seem pointless since all iOS devices so far have an accelerometer, Apple does provide this check. In future there may be devices that don't have an accelerometer. In fact, if you consider that the Apple TV (2nd generation) is running a version of iOS, this future may not be that far away. Who knows? (Apple, of course)

Where available (devices running iOS 4.0 and newer) KKInput will automatically use the Core Motion Framework to obtain the acceleration values. On devices running iOS 3.x and on iOS Simulator (for compatibility with iSimulate) KKInput falls back to using the UIAccelerometer methods and UIAcceleration values. This should not have a noticable affect on the quality of the acceleration values or the performance.

Accessing Acceleration Values

KKInput exposes acceleration values through the KKAcceleration class. Through KKAcceleration properties you get access to the raw, smoothed (low-pass filter) and instantaneous (high-pass filter) acceleration values.

```
KKInput* input = [KKInput sharedInput];
KKAcceleration* acceleration = input.acceleration;

CGPoint raw, smooth, instant;

// raw, unmodified accelerometer values
raw = CGPointMake(acceleration.rawX, acceleration.rawY);

// low-pass filtered (smoothed) values
smooth = CGPointMake(acceleration.smoothedX, acceleration.smoothedY);

// high-pass filtered (instantaneous) values
CGPoint instant = CGPointMake(acceleration.instantaneousX, acceleration.instantaneousY);
```

You can keep a weak reference of the KKAcceleration object. It will continue to be updated and isn't released during the lifetime of your app.

You do not have to consider device orientation. The acceleration values are already mapped so that the positive x axis always extends to the right, and the positive y axis always extends upwards regardless of the device orientation.

About Filtering

The `smoothed` properties use a low-pass filter. The low-pass filter cancels out sudden but short-lived acceleration spikes. The downside is that values react more slowly to change. This is usually ideal for controls like in "Tilt to Live".

This is the low-pass filter used in the current implementation:

```
-(void) applyLowPassFilter
{
  smoothedX = (rawX * filteringFactor) +
              (smoothedX * (1.0 - filteringFactor));
  smoothedY = (rawY * filteringFactor) +
              (smoothedY * (1.0 - filteringFactor));
  smoothedZ = (rawZ * filteringFactor) +
              (smoothedZ * (1.0 - filteringFactor));
}
```

The `instantaneous` properties use a high-pass filter. The high-pass filter pronounces the sudden but short-lived acceleration values, and is therefore the opposite of the low-pass filter. Instantaneous values can be used for detecting various shake gestures but their use in games is limited.

This is the high-pass filter used in the current implementation:

```
-(void) applyHighPassFilter
{
  instantaneousX = rawX - ((rawX * filteringFactor) +
                   (instantaneousX * (1.0 - filteringFactor)));
  instantaneousY = rawY - ((rawY * filteringFactor) +
                   (instantaneousY * (1.0 - filteringFactor)));
  instantaneousZ = rawZ - ((rawZ * filteringFactor) +
                   (instantaneousZ * (1.0 - filteringFactor)));
}
```

You can change the `filteringFactor` default value from `0.1f` to other values between `0.0f` and `1.0f` to make the filtering more or less pronounced. Both high and low pass filters use the same filteringFactor because the high-pass filter is basically the inverse of the low-pass filter. Normally you are only interested in either one, and if you're interested in both you'll want them to be "synchronized".

The filtering is applied only when you access the smoothed or instantaneous properties. The filtering algorithm is guaranteed to be run only once per frame, even if you access these properties multiple times.

KKInput Gyroscope

KKInput supports Gyroscope input on devices that support it.

- Gyroscope Availability
- Enabling Gyroscope Input
- Retrieving Gyroscope values

Gyroscope Availability

Devices that have a Gyroscope:

- iPhone 4, 4S and newer
- iPod Touch 4th generation and newer
- iPad 2 and newer

These devices **do not** have a Gyroscope:

- iPhone, iPhone 3G, iPhone 3GS
- iPod Touch 1st, 2nd and 3rd generation

- iPad
- iOS Simulator

To test if Gyroscope input is available on the current device:

```
if ([KKInput sharedInput].gyroAvailable) {...}
```

Enabling Gyroscope Input

To enable Gyroscope input:

```
[KKInput sharedInput].gyroActive = YES;
```

On devices that do not support Gyroscope this code has no effect. You should disable Gyroscope input whenever you don't need it to conserve battery power of the user's device.

Retrieving Gyroscope values

The Gyroscope values are wrapped in a KKRotationRate (class reference) object.

```
KKInput* input = [KKInput sharedInput];
KKRotationRate* rotationRate = input.rotationRate;
CGPoint rot = CGPointMake(rotationRate.x, rotationRate.y);
```

You can keep a weak reference to the KKRotationRate object. It is valid during the entire lifetime of your app and its values continue to be updated.

KKInput Device Motion

KKInput supports Device Motion input, which gives you access to user acceleration, gravity acceleration and gyroscope values all at once. With sensor fusion you also get access to the device orientation through roll/pitch/yaw Euler angles. You can also access the Core Motion CMAttitude object which contains the same orientation information as a rotationMatrix and a quaternion.

Device Motion Availability

Devices that support Device Motion:

- iPhone 4, 4S and newer
- iPod Touch 4th generation and newer
- iPad 2 and newer

These devices **do not** support Device Motion:

- iPhone, iPhone 3G, iPhone 3GS
- iPod Touch 1st, 2nd and 3rd generation
- iPad
- iOS Simulator

To test if Device Motion input is available on the current device:

```
if ([KKInput sharedInput].deviceMotionAvailable) {...}
```

Enabling Device Motion Input

To enable Device Motion input:

```
[KKInput sharedInput].deviceMotionActive = YES;
```

On devices that do not support Device Motion this code has no effect. You should disable Device Motion input whenever you don't need it to conserve battery power of the user's device.

You should not enable Device Motion and regular Gyroscope or Accelerometer input at the same time. This can result in incorrect motion values. You get access to Gyroscope and Accelerometer values through Device Motion.

Retrieving Device Motion values

The Device Motion values are wrapped in a KKDeviceMotion (class reference) object.

```
KKInput* input = [KKInput sharedInput];
KKDeviceMotion* motion = input.deviceMotion;

// sensor-fused acceleration values
KKAcceleration* userAcceleration = motion.acceleration;
KKAcceleration* gravityAcceleration = motion.gravity;

CGPoint userAcc, gravityAcc;
userAcc = CGPointMake(userAcceleration.rawX, userAcceleration.rawY);
gravityAcc = CGPointMake(gravityAcceleration.rawX, gravityAcceleration.rawY);

KKRotationRate* rotationRate = motion.rotationRate;
CGPoint rot = CGPointMake(rotationRate.x, rotationRate.y);

// sensor-fused device orientation
CGPoint attitude = CGPointMake(motion.pitch, motion.roll);

#if KK_PLATFORM_IOS
CMAttitude* cmAttitude = (CMAttitude*)motion.attitude;
CMRotationMatrix rotMatrix = cmAttitude.rotationMatrix;
CMQuaternion quaternion = cmAttitude.quaternion;
#endif
```

With Device Motion you get access to two types of acceleration values: userAcceleration and gravityAcceleration. If you would add both value sets together you would get the total acceleration of the device. With `userAcceleration` you get only the user component of the acceleration, minus the constant gravitational acceleration. Likewise `gravityAcceleration` gives you only the gravitational component of the acceleration, minus the user acceleration.

For both userAcceleration and gravityAcceleration you will want to access the "raw" values, since these acceleration values provide a much smoother input than non-DeviceMotion acceleration values thanks to sensor-fusion with the Gyroscope values.

You also get access to the Gyroscope KKRotationRate object through Device Motion. The values are the same as described in KKInput Gyroscope.

KKDeviceMotion exposes the commonly used pitch, roll and yaw attributes (also known as Euler angles). These values provide you with the current orientation of the device. Alternatively, if you need and are comfortable with rotation Matrices and Quaternions, you can get access to the CMAttitude object and thus the rotationMatrix and quaternion properties. This however would be iOS-specific code and needs to be wrapped in `#if KK_PLATFORM_IOS .. #endif` in case your app should also have a Mac OS version.

You can keep a weak reference to the KKDeviceMotion object. It is valid during the entire lifetime of your app and its values continue to be updated.

KKInput Keyboard

KKInput supports testing the state of the Keyboard keys. The keyboard is accessed with virtual key codes in a locale-independent manner.

- About Locale Independent Keyboard Input
- Any Key Down or Up
- Testing for a particular Key press
- Key presses with modifiers
- Keyboard Input References

About Locale Independent Keyboard Input

It is a common mistake among beginning game developers to assume that testing for the key character "/" will work correctly on all keyboards.

Assume the following (bad) example code:

```
-(void) ccKeyDown:(NSEvent*)event
{
    // BAD EXAMPLE!!!
    if ([event characters] == @"/")
    {
        // code for shooting here
    }
}
```

This works fine on a US Keyboard. You press the key that's left to the Right Shift key. However, most european players would have to press Shift+7 in order to shoot. That's because the key to the left of the Right Shift key prints a different character, for example on German keyboards it will print the "-" character. Clearly you'll want to avoid that!

Any keyboard character can be mapped to a different key or key combination depending on the user's locale. Just about every nationality has its own keyboard layout! That's why you don't want to test for the characters the keyboard sends to the operating system, you'll want to test specifically for a key's KeyCode that uniquely identifies each key on all keyboards.

Games rely on these so-called virtual keycodes to treat the Keyboard as an input device where each key has a unique key code that's the same for all keyboards and all locale settings. Games don't care what's printed on the key to the left of the Right Shift key, a game wants to ensure that they keyboard to game actions mapping is the same for all locales.

Any Key Down or Up

The simplest way to check for Keyboard input is obviously the "any" key. You know, that particular key one always has to press to continue, but that's never been on any keyboard.

Joking aside, if all you need to test if a key is pressed, either for simple menus or for a "pre-screening" if it's worthwhile to perform more thorough keyboard input checks, use one of these methods (properties actually):

```
KKInput* input = [KKInput sharedInput];
if (input.isAnyKeyDown) {...}
if (input.isAnyKeyDownThisFrame) {...}
if (input.isAnyKeyUpThisFrame) {...}
```

In case you're wondering why there's no `isAnyKeyUp`, that's because it's not very meaningful. Think about it. A common keyboard has around and often times over 100 keys. One of those keys is very likely going to be "up" (as in: not pressed) until the day humans developed tentacle limbs.

Testing for a particular Key press

KKInput has these methods to test for key down/up states by providing a virtual keycode for the desired key:

```
KKInput* input = [KKInput sharedInput];
if ([input isKeyDown:KKKeyCode_Space]) {...}
if ([input isKeyDownThisFrame:KKKeyCode_Return) {...}
if ([input isKeyUp:KKKeyCode_W) {...}
if ([input isKeyUpThisFrame:KKKeyCode_RightArrow) {...}
```

If you want to test for a combination of keys simply && (logical AND) the queries together. Likewise, if you need to respond to alternate keys doing the same thing use || (logical OR):

```
KKInput* input = [KKInput sharedInput];

// key combination:
if ([input isKeyUpThisFrame:KKKeyCode_Slash] &&
    [input isKeyUpThisFrame:KKKeyCode_Backslash]) {...}

// alternate keys:
if ([input isKeyDown:KKKeyCode_S] ||
    [input isKeyDown:KKKeyCode_DownArrow]) {...}
```

Adding key combos and alternate keys as an array of keycodes may be available in the future, please refer to the status of this task.

Key presses with modifiers

Modifiers are those "special" keys like Command, Control, Shift and others, defined in the KKModifierFlag enum.

You have two ways of checking whether a key **and** one or more modifier keys have been pressed down. The Modifier keys have also virtual key codes, so you can just and them together as in the first part of the following code example. The other way is to use the modifier flags enum. I'll explain the major issue with the first approach after the code - maybe you can already guess it?

```
KKInput* input = [KKInput sharedInput];

// #1) Testing for: Control+Command+Shift+C
if ([input isKeyDownThisFrame:KKKeyCode_C] &&
    [input isKeyDown:KKKeyCode_Control] &&
    [input isKeyDown:KKKeyCode_Command] &&
    [input isKeyDown:KKKeyCode_Shift]) {...}

// #2) Testing for: Control+Command+Shift+C with Modifier flags
if ([input isKeyDownThisFrame:KKKeyCode_C
                modifierFlags:KKModifierControlKeyMask |
                              KKModifierCommandKeyMask |
                              KKModifierShiftKeyMask]) {...}

// examples for all methods which take modifierFlags:
if ([input isKeyDown:KKKeyCode_A
                modifierFlags:KKModifierShiftKeyMask]) {...}
if ([input isKeyDownThisFrame:KKKeyCode_B
                modifierFlags:KKModifierControlKeyMask]) {...}
if ([input isKeyUp:KKKeyCode_C
                modifierFlags:KKModifierCommandKeyMask]) {...}
if ([input isKeyUpThisFrame:KKKeyCode_D
                modifierFlags:KKModifierFunctionKeyMask]) {...}
```

The problem with #1) is in particular related to the "ThisFrame" check. What you want to achieve is to test whether the modifier keys (Control, Command and Shift) are held down when the "C" key is pressed "ThisFrame". The first method is prone to human error introducing a subtle, hard to notice but fatal bug: if you also checked one of the modifiers or a second key to be pressed down "ThisFrame" the user would have to press both these keys down in the same frame. It is almost impossible to press down two or more keys down at the same time with a time span of 0.1667 seconds (one 60th of a second).

That's why the #2) approach is preferred, it assures that the test (key down/up, this frame or not) is performed only on the defining key and not one of the modifier keys. It also requires fewer method calls and is a bit shorter to write because you can | (bitwise OR) the modifier key masks together. Approach #2) will check if the "C" key was pressed down "ThisFrame", and then verifies if the modifier keys have also been held down at the same time.

Keyboard Input References

Kobold2D Virtual KeyCodes Reference with KeyCode values printed in hex.

For more details about Mac Keyboard Input processing please refer to this blog post.

Here's a handy KeyCodes App that prints out all the information about the character when you press it.

Image of Mac Keyboard with virtual KeyCodes printed on each key. Click image for high-res version. Image courtesy of Peter Hosey.

Figure C-2 Apple Extended Keyboard II (domestic layout)
From Inside Macintosh: Text

KKInput Mouse

KKInput supports mouse input to get the mouse location, button and scroll wheel states.

- Mouse Location
- Any Mouse Button Down/Up
- Specific Mouse Button Down/Up
- Specific Mouse Button Down/Up with Modifier flags
- Testing for Mouse Button Double Clicks

Mouse Location

By default mouse locations are updated only when one of the mouse button or mouse wheel states change. To track the mouse location in every frame you have to enable the mouse moved events. This is the same as setting `AcceptsMouseMovedEvents = YES` in *config.lua*:

```
[KKInput sharedInput].acceptsMouseMovedEvents = YES;
```

You get access to three different mouse locations via KKInput:

```
KKInput* input = [KKInput sharedInput];
CGPoint mousePos = input.mouseLocation;
CGPoint previous = input.previousMouseLocation;
CGPoint velocity = input.mouseLocationDelta;
```

The `mouseLocationDelta` is nothing more than the difference between the `mouseLocation` and the `previousMouseLocation`. In other words it's the velocity vector of the mouse.

Any Mouse Button Down/Up

If you need to do a quick & dirty check for all mouse buttons together, you can use these convenience methods:

```
KKInput* input = [KKInput sharedInput];
if (input.isAnyMouseButtonDown) {...}
if (input.isAnyMouseButtonDownThisFrame) {...}
if (input.isAnyMouseButtonUpThisFrame) {...}
```

There is no `isAnyMouseButtonUp` property because it makes little to no sense.

Specific Mouse Button Down/Up

You can test for specific mouse buttons by passing a KKMouseButtonCode to the following methods:

```
KKInput* input = [KKInput sharedInput];
if ([input isMouseButtonDown:KKMouseButtonLeft]) {...}
if ([input isMouseButtonDownThisFrame:KKMouseButtonLeft]) {...}
if ([input isMouseButtonUp:KKMouseButtonOther + 1]) {...}
if ([input isMouseButtonUpThisFrame:KKMouseButtonRight]) {...}
```

You can test for "other" buttons with the `KKMouseButtonOther` code. The left and right mouse buttons are buttons 1 and 2. `KKMouseButtonOther` equals mouse button number 3, and incrementing the other button as in `KKMouseButtonOther + 1` refers to the mouse button 3 + n, in this case the 4th mouse button.

Most mice report only two buttons on Mac OS. Even mice with more than two buttons often only report two buttons to Mac OS, most likely because of mouse driver issues (not installed, no Mac driver, driver broken, etc). Do not rely on any of the "other" mouse buttons to be available. There is also currently no way of testing how many buttons the connected mouse supports. Thus it is recommended to design your game to be playable with at most the left and right mouse buttons.

Specific Mouse Button Down/Up with Modifier flags

In some scenarios you will want to use modifier key flags together with mouse button presses. For example to test for Control+LeftMouseClick and similar events. Instead of combining the keyboard modifier flags with the mouse button presses, you can test for such a combination with these methods:

```
KKInput* input = [KKInput sharedInput];
if ([input isMouseButtonDown:KKMouseButtonLeft
                modifierFlags:KKModifierControlKeyMask]) {...}
if ([input isMouseButtonDownThisFrame:KKMouseButtonLeft
                modifierFlags:KKModifierShiftKeyMask |
                              KKModifierCommandKeyMask]) {...}
```

There are no "mouse button up" methods with modifierFlags for technical reasons.

Testing for Mouse Button Double Clicks

You use the same methods as before, but you'll pass the special "double click" mouse button codes:

```
KKInput* input = [KKInput sharedInput];
if ([input isMouseButtonDown:KKMouseButtonDoubleClickLeft]) {...}
if ([input isMouseButtonDownThisFrame:KKMouseButtonDoubleClickLeft]) {...}
if ([input isMouseButtonUp:KKMouseButtonDoubleClickOther + 1]) {...}
if ([input isMouseButtonUpThisFrame:KKMouseButtonDoubleClickRight]) {...}

if ([input isMouseButtonDown:KKMouseButtonDoubleClickLeft
                modifierFlags:KKModifierControlKeyMask]) {...}
if ([input isMouseButtonDownThisFrame:KKMouseButtonDoubleClickRight
                modifierFlags:KKModifierShiftKeyMask |
                              KKModifierCommandKeyMask]) {...}
```

KKInput currently doesn't give you access to the mouse button click count. If you need the click count, please request it.

Working with Automatic Reference Counting (ARC)

This guide will introduce you to automatic reference counting (ARC). ARC is fully supported by Kobold2D and enabled by default in all template projects. If you start a new Kobold2D project, you're already using ARC. Of course you can disable ARC but you shouldn't really need to.

This guide also explains what you need to be aware of when interfacing with C or C++ libraries (ie. Chipmunk, Box2D) in an ARC-enabled project.

Kobold2D has automatic reference counting (ARC) enabled by default.

What is ARC?

How Apple describes ARC:

Automatic Reference Counting (ARC) for Objective-C makes memory management the job of the compiler. By enabling ARC with the new Apple LLVM compiler, you will never need to type retain or release again, dramatically simplifying the development process, while reducing crashes and memory leaks. The compiler has a complete understanding of your objects, and releases each object the instant it is no longer used, so apps run as fast as ever, with predictable, smooth performance.

Kobold2D ARC Documentation

- ARC Requirements for Development and Deployment — ARC has several requirements both for development and for deployment, ie on which devices and OS versions an ARC app runs on.
- What you need to know about ARC — This article tells you what you must know about ARC to be able to work with it, or convert an existing Objective-C project to ARC.
- ARC with C or C++ code (Chipmunk, Box2D) — All is fine and well with ARC in Objective-C land. However, as soon as you also work with C or C++ code — such as the Chipmunk or Box2D libraries, or the Core Foundation framework — you may find yourself facing compiler errors.
- Enabling ARC in legacy Kobold2D projects — If your project is based on Kobold2D Preview 5 or earlier, you will have to perform the following steps to enable ARC in your project.
- Disabling Automatic Reference Counting — If you want to deploy Kobold2D apps to devices running iOS 3 you must disable ARC.

More about ARC

If you want to understand in layman's terms how ARC works, you should read this question on stackoverflow.com: How does the new automatic reference counting mechanism work?

If you're totally into the nitty-gritty technical details of ARC you might want to read the LLVM compiler's Automatic Reference Counting documentation.

Mike Ash has summarized how ARC works with many practical examples.

On a higher level, Wikipedia tells us what reference counting actually is and how it works.

ARC Requirements for Development and Deployment

ARC has several requirements both for development and for deployment, ie on which devices and OS versions an ARC app runs on.

- Development Requirements
- iOS Device Deployment Requirements
- Mac OS X Deployment Requirements

Development Requirements

- Mac OS X 10.6 Snow Leopard and newer for ARC-enabled **iOS** Apps
- Mac OS X 10.7 Lion and newer for ARC-enabled **Mac** Apps
- Xcode 4.2 and newer
- Apple LLVM compiler 3.0 (default compiler in Xcode 4.2)

You can not create ARC apps with Xcode 4.1 or earlier because the Apple LLVM compiler 3.0 is shipped only with Xcode 4.2 and the iOS 5 SDK. You can not create Mac ARC apps on Snow Leopard because you need the Mac OS X 10.7 SDK to build ARC-enabled Mac apps.

If you are using Xcode 4.1 or earlier then the Kobold2D projects will still work. Kobold2D detects if ARC is unavailable and makes sure the code compiles and runs without ARC.

iOS Device Deployment Requirements

- iOS 4.0 or newer (2nd Generation Devices or newer)
- weak references: iOS 5.0 or newer (3rd Generation Devices or newer)

In other words, ARC enabled apps run on all iOS devices with at least iOS 4.0. However, weak references are only available on devices running iOS 5. If you want to use the `weak` @property modifier or the `__weak` keyword you should set your app's deployment target to iOS 5.

This results in the following devices capable of running ARC enabled apps:

- Devices capable of running iOS 4.x but not iOS 5:
    - iPhone 3G, iPod Touch 2nd generation
- Devices capable of running iOS 5:
    - iPhone 3GS, iPhone 4, iPhone 4S
    - iPod Touch: 3rd & 4th Generation
    - iPad, iPad 2

Mac OS X Deployment Requirements

ARC is available only for 64-Bit Mac apps!

- Mac OS X 10.6 (Snow Leopard, **64-Bit**) or newer
- weak references: Mac OS X 10.7 (Lion) or newer

Mac apps with ARC enabled are compatible with Mac OS X 10.6 Snow Leopard.
32-Bit ARC apps are not supported. You can only build 64-Bit apps with ARC enabled.

Lion is a 64-Bit system, so every Mac that runs Lion will be able to run 64-Bit apps. Snow Leopard however will only run 64-Bit apps on Macs that support it (Core 2 Duo processor and newer).

What you need to know about ARC

This article tells you what you must know about ARC to be able to work with it, or convert an existing Objective-C project to ARC.

Note that Apple has an excellent document Transitioning to ARC Release Notes where they describe in greater detail how to work with ARC.

- ARC Do's, Let Do's and Dont's
- ARC properties
- C structs storing objects

ARC Do's, Let Do's and Dont's

What you do need to know is what you can't and shouldn't do, because ARC takes care of memory management and imposes a few restrictions. In particular, follow these rules when writing ARC code or updating non-ARC code to be compatible with ARC:

- Remove any calls to:
    - `retain`
    - `retainCount`
    - `release`
    - `autorelease`
    - `dealloc` (specifically: `[super dealloc]`)
- Do not override these methods in your class:
    - `retain`
    - `retainCount`
    - `release`
    - `autorelease`
- Do not give a @property a name that begins with "new"
- Replace NSAutoreleasePool with the @autoreleasepool compiler directive

It is legal and sometimes required to override the `-(void) dealloc` method, for example to release the memory of C or C++ objects. But you can not call `[super dealloc]`, adding this call to your `dealloc` method is a habit that you'll have to let go.

In C structs you should not store object pointers `id`. Instead, create an Objective-C class in place of the struct. If necessary, mark the instance variables of that class @public so that access to the instance variables is just as direct and fast as with a C struct. See below for an alternative solution, in particular if you can't easily change the C struct.

You can not freely cast between `id` — or any object pointer like for example `NSObject*` — and `void*`. You need to "bridge" the cast to tell ARC about the lifetime of the object. More on that the section about interfacing with C and C++ code.

ARC properties

Since you can't call or implement the retain method anymore, it is good practice to use the appropriate ARC keywords for properties as well. The modifiers `strong` and `weak` are synonymous for `retain` and `assign`.

`weak` is only available on iOS 5 and Mac OS X 10.7 (Lion), so if you want to make use of weak references you must set the deployment target accordingly.

```
// synonymous with: @property (retain) CCSprite *mySprite;
@property (strong) CCSprite *mySprite;

// synonymous with: @property (assign) CCSprite *mySprite;
// but if mySprite is deallocated, the property value is set to nil
// instead of remaining as a dangling pointer
@property (weak) CCSprite *mySprite;
```

It is still legal under ARC to use the `retain` and `assign` property modifiers. The compiler will assume `retain` to mean `strong` and `assign` to mean `__unsafe_unretained` (not: `weak`).

You can not accidentally create an ARC app that uses a weak reference without explicitly specifying the `weak` @property modifier or the `__weak` keyword.

C structs storing objects

If you have C structs in your code which stores objects (id, NSObject* or similar types), you'll get an error with ARC enabled:

```
typedef struct
{
 id anObject; // ERROR
} aStruct;
```

Error: `ARC forbids Objective-C objects in structs or unions`

The solution is to tell ARC that the object stored in the struct should simply be assigned. ARC rightfully declares this as "unsafe" because it can lead to dangling pointers, but if your code has worked so far, you'll be fine by adding the `__unsafe_unretained` keyword before the object's type:

```
typedef struct
{
 __unsafe_unretained id anObject;
} aStruct;
```

ARC with C or C++ code (Chipmunk, Box2D)

All is fine and well with ARC in Objective-C land. However, as soon as you also work with C or C++ code — such as the Chipmunk or Box2D libraries, or the Core Foundation framework — you may find yourself facing compiler errors.

Fortunately, it's half as bad if you understand why the compiler keeps bugging you.

- Errors you'll see when casting id to void* and vice versa
- What is a "bridged cast"?
- Bridge casts for Core Foundation
- More about bridge casts
- Box2D & Chipmunk real-world examples

Errors you'll see when casting id to void* and vice versa

Most errors when working with C/C++ code has to do with the fact that you can't freely cast `id` to `void*` and vice versa. For example, the below example won't work:

```
NSObject* myObject = [[NSObject alloc] init];
void* myObjectPointer = myObject; // ERROR

void* myObjectPointer2 = NULL;
NSObject* myObject2 = myObjectPointer2; // ERROR
```

You'll get an error like this one:
`error: cannot initialize a variable of type '...' with an lvalue of type '...'`

The types in the error message mentioned may differ but you'll see a `__strong` keyword thrown in for good measure. Well, you might think this can be fixed by properly casting to the correct types:

```
NSObject* myObject = [[NSObject alloc] init];
void* myObjectPointer = (void*)myObject; // ERROR, still ...

void* myObjectPointer2 = NULL;
NSObject* myObject2 = (NSObject*)myObjectPointer2; // ERROR, still ...
```

This time the error is different, each of the two casts above generates a different message:

1: `Cast of Objective-C pointer type 'NSObject *' to C pointer type 'void *' requires a bridged cast`
2: `Cast of C pointer type 'void *' to Objective-C pointer type 'NSObject *' requires a bridged cast`

What is a "bridged cast"?

ARC adds the keywords `__bridge`, `__bridge_transfer` and `__bridge_retained` which must be used when casting. Here's the above example with a "bridged cast" that fixes the compile errors:

```
NSObject* myObject = [[NSObject alloc] init];
void* myObjectPointer = (__bridge void*)myObject;

void* myObjectPointer2 = NULL;
NSObject* myObject2 = (__bridge NSObject*)myObjectPointer2;
```

The use of `__bridge` in the cast tells the compiler that the object's lifetime doesn't change, and that you understand that if the object's memory is released it may result in a dangling pointer and a potential crash. This is the same behavior to non-ARC code, but ARC really wants to make sure you understand the implications and potential dangers. In ARC terms, you specify that the cast creates a "unsafe unretained" reference.

Bridge casts for Core Foundation

When interfacing with Core Foundation you may want to use `__bridge_retain` to tell ARC to take on ownership of the object (retain it). Likewise, `__bridge_transfer` tells ARC to release the object after the expression.

Do not apply retain or transfer bridge casts to objects that are not Core Foundation (CF) objects!

More about bridge casts

For more details about bridged casts read the section about Bridged Casts and the follow-up section Conversion of retainable object pointers in the LLVM compiler documentation.

Box2D & Chipmunk real-world examples

```
CCSprite* aSprite = [CCSprite spriteWithFile:@"image.png"];


// Assigning sprite to Box2D userdata
b2BodyDef bodyDef;
bodyDef.userData = (__bridge void*)aSprite;

// Getting the userdata sprite from Box2D
CCSprite* sprite = (__bridge CCSprite*)body->GetUserData();


// Assigning sprite to Chipmunk userdata
cpShape* shape = cpPolyShapeNew(staticBody, numVertices, vertices, offset);
shape->data = (__bridge void*)aSprite;

// Getting the userdata sprite from Chipmunk
CCSprite* sprite = (__bridge CCSprite*)shape->data;
```

Enabling ARC in legacy Kobold2D projects

If your project is based on Kobold2D Preview 5 or earlier, you will have to perform the following steps to enable ARC in your project.

- Update Build Settings
- Import kkARCSupport.h in Prefix header file
- The KK_ARC_ENABLED macro
- The ARC Conversion Tool provided by Xcode

Update Build Settings

Open your project's workspace in Xcode. Browse to the *BuildSettings* group in the Project Navigator and locate the files *BuildSettings-iOS.xcconfig* and *BuildSettings-Mac.xcconfig*. To both files (if existing) add the following line at the end of the file:

```
// Enable automatic reference counting (ARC)
CLANG_ENABLE_OBJC_ARC = YES
```

Import kkARCSupport.h in Prefix header file

Browse to the group *ProjectFiles/Supporting Files* and locate the files *Prefix-iOS.pch* and *Prefix-Mac.pch*. To both files add the *kkARCSupport.h* header file below the *kobold2d.h* header file:

```
// Kobold2D headers
#import "kobold2d.h"
#import "kkARCSupport.h"
```

This ensures that libraries such as cocos2d compile error-free with ARC enabled.

The KK_ARC_ENABLED macro

The *kkARCSupport.h* header file also makes the `KK_ARC_ENABLED` macro available, with which you can conditionally compile code if ARC is either enabled or disabled/unavailable.

You will probably not need to use this macro, but you'll find it used in many of the Kobold2D template projects. It is used to keep the Kobold2D template projects compatible with non-ARC compilers (Xcode 4.1 or earlier) respectively for users who want or need to disable ARC. For example:

```
#ifdef KK_ARC_ENABLED
 CCLOG(@"ARC is enabled");
#else
 CCLOG(@"ARC is either not available or not enabled");
#endif
```

The ARC Conversion Tool provided by Xcode

In Xcode you can convert any Objective-C project to ARC with the built-in ARC conversion tool. With your project open you should select **Edit -> Refactor -> Convert to Objective-C ARC...** from the menu to start the conversion process.

Most of the time the conversion tool gets things right, but not always. If it fails for some reason please leave a message in the Kobold2D forum and include the code sample where Xcode reports a compile error.

Usually the issues are simple to fix, here are some guidelines:

- remove all calls to: `release`, `autorelease`, `retain`
- remove `super dealloc` and any other calls to `dealloc`
- in C structs prefix all `id` variables with `__unsafe_unretained`
- when casting `id` or other Objective-C class instance pointers from or to void*, cast the pointer with the `__bridge` keyword like so: `id object = (__bridge id)somePointer` respectively `void* pointer = (__bridge void*)someObject`

Disabling Automatic Reference Counting

If you want to deploy Kobold2D apps to devices running iOS 3 you must disable ARC.

- Disable ARC in iOS Projects
- Disable ARC in Mac OS X Projects

Disable ARC in iOS Projects

In your project's Build Settings, ensure that:

- `Objective-C Automatic Reference Counting` is set to NO
- `iOS Deployment Target` is set to iOS 3.1

You should not target iOS 3.0 since the display link director will be unavailable on iOS 3.0. For that reason Kobold2D officially supports only iOS 3.1 and newer.

Various reports indicate that iOS 3 usage is definitely below 5%, and in this report nearing 2%. Most developers agree that it is no longer worthwhile to support devices running iOS 3. Please consider that before deciding against using ARC in your project to support a dying operating system version.

Disable ARC in Mac OS X Projects

In your project's Build Settings, ensure that:

- `Objective-C Automatic Reference Counting` is set to NO

In *BuildSettings-Mac.xcconfig* in the *BuildSettings* group ensure that:

- the setting `CLANG_ENABLE_OBJC_ARC[arch=x86_64]` is set to NO or commented out or removed

On Mac OS X you gain no benefit from disabling ARC. With or without ARC the minimum deployment target is Mac OS X 10.6 (Snow Leopard). Under Mac OS X 10.7 (Lion) all apps are 64-Bit anyway, so the "Lion only supports ARC for 64-Bit apps" restriction is not really a restriction.

Kobold2D Reference Documentation

Online Reference Manuals

Cocos2D Resources

Cocos3D Programming Guide

CocosDenshion FAQ
CocosDenshion CookBook
ObjectAL in a Nutshell

Box2D User Manual
Chipmunk Documentation

Lua Reference Manual

Online API References

- Box2D API Reference
- Chipmunk API Reference
- Chipmunk SpaceManager API Reference
- Cocos2D API Reference (iOS)
- Cocos2D API Reference (Mac OS)
- Cocos2D Extensions API Reference (iOS)
- Cocos2D Extensions API Reference (Mac OS)
- CocosDenshion API Reference (iOS)
- CocosDenshion API Reference (Mac OS)
- Kobold2D API Reference (iOS)
- Kobold2D API Reference (Mac OS)
- ObjectAL API Reference (iOS)
- SneakyInput API Reference

API References

Get help in Xcode!
Kobold2D installs the API References into Xcode's help system. In Xcode, choose **Help -> Xcode Help**, then switch to the **Editor -> Explore Documentation** panel.

Kobold2D 2.0 API References

- Box2D API Reference
- Chipmunk API Reference
- Chipmunk SpaceManager API Reference
- Cocos2D API Reference (iOS)
- Cocos2D API Reference (Mac OS)
- Cocos2D Extensions API Reference (iOS)
- Cocos2D Extensions API Reference (Mac OS)
- CocosDenshion API Reference (iOS)
- CocosDenshion API Reference (Mac OS)
- Kobold2D API Reference (iOS)
- Kobold2D API Reference (Mac OS)
- ObjectAL API Reference (iOS)
- SneakyInput API Reference

Looking for Earlier Version API References?

Go to the API References archive.

These API References are not exclusive to Kobold2D. Even if you're not a Kobold2D user, you can refer to the information in the API References for the libraries you use. The API References are usually for the most up to date version of the library.

Box2D API Reference

Open in new window ...

Chipmunk API Reference

Open in new window ...

Chipmunk SpaceManager API Reference

Open in new window ...

Cocos2D API Reference (iOS)

Open in new window ...

Cocos2D API Reference (Mac OS)

Open in new window ...

Cocos2D Extensions API Reference (iOS)

Open in new window ...

Cocos2D Extensions API Reference (Mac OS)

Open in new window ...

CocosDenshion API Reference (iOS)

Open in new window ...

CocosDenshion API Reference (Mac OS)

Open in new window ...

Kobold2D API Reference (iOS)

Open in new window ...

Kobold2D API Reference (Mac OS)

Open in new window ...

ObjectAL API Reference (iOS)

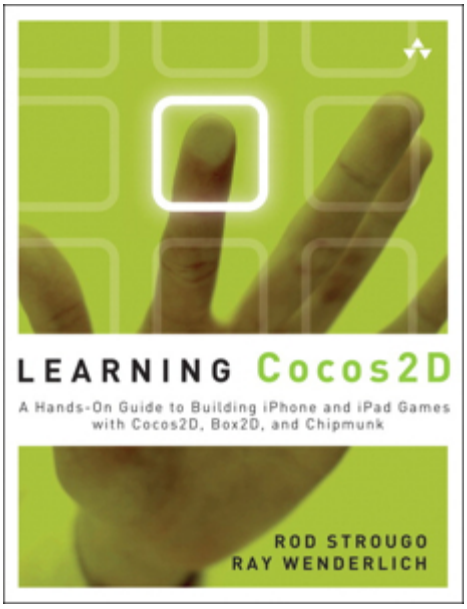
Open in new window ...

SneakyInput API Reference

Open in new window ...

Cocos2D Resources

Learn iPhone and iPad Cocos2D Game Development Learning Cocos2D


LEARNING Cocos2D
A Hands-On Guide to Building iPhone and iPad Games with Cocos2D, Box2D, and Chipmunk
ROD STROUGO
RAY WENDERLICH

Authors:	Steffen Itterheim, Andreas Löw
Published:	November 2010 (1st Edition) October 2011 (2nd Edition)
Cocos2D Version:	v0.99 (1st Ed) — v1.0 (2nd Ed)
Source Code:	Free Download: v0.99 — v1.0
Websites:	Book Website Apress Website

Authors:	Rod Strougo, Ray Wenderlich
Published:	July 2011 (1st Edition)
Cocos2D Version:	v1.0 (1st Ed)
Source Code:	Free Download: v1.0
Websites:	Book Website Addison-Wesley Website

Why are these books recommended?
The above books are up to date, accurate, easy to follow and cover a broad scope of topics — including UIKit integration, physics engines, tilemaps, performance optimizations and development tools.

Cocos2D Tutorials

There are plenty of tutorials for Cocos2D on the net. But only a few tutorial authors stand out from the crowd:

- Ray Wenderlich — Cocos2D & iOS Tutorials
- Mohammad Azam — Cocos2D & Mobile Development Tutorials and Videos
- SDKTutor — Cocos2D Tutorial Videos

Cocos2D Podcast

The Cocos2D Podcast is hosted by Mohammad Azam and Steffen Itterheim. They're often joined by guest speakers from the community to talk about Cocos2D, Game Development, and other things of interest.

The Cocos2D Podcast is also available on iTunes.

Cocos2D Wiki

Then there's the official Cocos2D Documentation. You can learn a few bits and pieces from it but unfortunately it doesn't teach you Cocos2D in its entirety. That's a big blank which the books above filled in.

Lua Reference Manual

Open in new window ...

Kobold2D Contributor's Guide

Contributing

How to become a Kobold2D Contributor
Kobold2D Folder Structure Explained

Step by Step

Creating a new Project Template for Kobold2D
Adding a new Xcode File Template to Kobold2D
Adding a new Static Library to Kobold2D

Guidelines

Managing Kobold2D Build Settings
Guidelines for Kobold2D Development
Guidelines for Commercial Add-On Products

Kobold2D Build Status

The Build Status will be updated within minutes after a source code modification and indicate whether the code was built successfully or not.

Build Status not yet implemented

Kobold2D FAQ

- Kobold2D General FAQ
- Kobold2D Xcode FAQ
- Kobold2D Technical FAQ
- Kobold2D Cocos2D FAQ
- Kobold2D Wax & Lua FAQ
- Kobold2D Audio FAQ
- Kobold2D Physics FAQ
- Kobold2D Cocos3D FAQ

Kobold2D General FAQ

- Why was Kobold2D created?
- Why isn't Kobold2D a contribution to cocos2d-iphone?
- Should I try Cocos2D first before switching to Kobold2D?
- Do I need lots of Cocos2D experience to start with Kobold2D?
- Do all the Cocos2D tutorials and books work with Kobold2D?
- When was Kobold2D created?
- What does Kobold2D cost?
- Can I donate to Kobold2D?
- Why does Kobold2D consume 400 MB disk space?
- How can I reduce the installed size of Kobold2D?
- How can I install Kobold2D to a custom folder?
- How can I uninstall Kobold2D?

Kobold2D Xcode FAQ

- Why does Kobold2D build successfully but won't run?
- Why do I get "ld .. failed with exit code 1"?
- Is Kobold2D compatible with Xcode 3?
- Does Kobold2D work with the latest beta SDK?
- How can I create a blank Kobold2D project?
- Can I add the Kobold2D.xcodeproj to my project?
- Can I work in a copy of the Kobold2D Workspace?
- Why does Xcode crash after installing Kobold2D or the Xcode Help files?
- How can I copy the Build Log in Xcode?

Kobold2D Technical FAQ

- Can Kobold2D Projects be located in a custom folder?
- Where does the 'Network Connections' dialog come from?
- How to disable iSimulate?
- Why is the AppDelegate class empty?
- Are Kobold2D apps larger than pure cocos2d-iphone apps?
- How to fix compile errors with CocosBuilder?
- Why does Kobold2D compile libraries I don't use?

Kobold2D Cocos2D FAQ

- Is Kobold2D compatible with cocos2d-iphone?
- Is Kobold2D updated when cocos2d-iphone is?
- Will installing Kobold2D affect my Cocos2D Projects?

Kobold2D Wax & Lua FAQ

- What's the difference between Wax and Corona SDK?
- Can I write game logic in Lua with Kobold2D?

Kobold2D Audio FAQ

- Which audio engine is better - CocosDenshion or ObjectAL?

Kobold2D Physics FAQ

- Which Physics Engine is the best?

Kobold2D Cocos3D FAQ

- Is there a Mac OS version of Cocos3D?

Kobold2D General FAQ

- Why was Kobold2D created?
- Why isn't Kobold2D a contribution to cocos2d-iphone?

- Should I try Cocos2D first before switching to Kobold2D?
- Do I need lots of Cocos2D experience to start with Kobold2D?
- Do all the Cocos2D tutorials and books work with Kobold2D?
- When was Kobold2D created?
- What does Kobold2D cost?
- Can I donate to Kobold2D?
- Why does Kobold2D consume 400 MB disk space?
- How can I reduce the installed size of Kobold2D?
- How can I install Kobold2D to a custom folder?
- How can I uninstall Kobold2D?

Why was Kobold2D created?

Oh, so many reasons ... here's a short excerpt of some key points.

Slow Progress

It took Cocos2D to progress from v0.99 to v1.0 from January 2010 to July 2011, with very few notable features added. The biggest changes were mandatory, like iPad and Retina support, and released only several months after the devices appeared on the market.

Cocos2D expressly focuses on keeping the core small, which is good on one side (stable, reliable API) but bad in that it means, for example, no Game Center integration and no high-level gameplay features.

Zynga

Zynga hired two key developers of Cocos2D. That means they no longer work full-time on cocos2d-iphone, or if they do so internally, their work is primarily for Zynga's purposes and we don't know if we actually see all the progress published to the open source version.

It is likely that at the very least those features which would benefit from being beta tested by a large userbase, such as OpenGL ES 2.0 or device-specific support, will be published.

Zynga claims to support Open Source software development but there are very few examples where they actually did so. Except for the Zynga copyright notices in the cocos2d-iphone source code files there are no indicators to this date (March 2012) that Zynga is actively involved in the development of cocos2d-iphone, or plans to involve themselves more than they really need to.

Case in point being that progress of cocos2d-iphone has slowed even more despite a multi-billion-dollar company backing it up. At the same time other projects (v1.x codebase & cocos2d-iphone-extensions) have been outsourced to volunteers from the community. That speaks volumes about Zynga's intentions towards contributing back to the open source community.

Lack of Vision

Where is cocos2d going?

There's a roadmap that's been mostly unchanged for the past two years. Most features are of technical nature, ie platform support, NSCoding, ARC, multiple EAGL Views.

Nothing regarding actual gameplay or convenience features frequently requested by developers. Pathfinding, AI, collision detection, scripting, networking, object pooling, none of that.

Sense of Urgency

Game Engines that compete with Cocos2D continue to grow at an incredible pace. Corona SDK, Cocos2D-X, Unity, Sparrow Framework, SIO 2 — they're all growing their user base and are becoming better products doing so.

Those other engines also offer direct-to-costumer support and documentation, and they actively work on implementing features that ease gameplay programming.

More and more beginning iOS and Mac developers wish to take advantage of iOS and Mac OS features in their games, without having to learn and understand Apple's APIs and the nitty-gritty details of the platform hardware, and their differences. Cocos2D barely helps you in that area.

Javascript

There are plans to add Javascript support to Cocos2D. Not unsurprisingly Zynga employs hundreds of Javascript developers.

The language of choice for the majority of game developers is of course Lua. Whereas Javascript closely resembles the C programming language, Lua is easier to learn for beginners and Flash developers as Corona SDK and other Lua engines have proven.

In Summary

Cocos2D needs a refresher, and it needs to grow to support the ever increasing number of beginning iOS & Mac developers. Kobold2D aims to be just that, easier for the beginners, more powerful for advanced users, highly optimized for productivity.

Why isn't Kobold2D a contribution to cocos2d-iphone?

Kobold2D is not and can not be just a contribution to cocos2d-iphone. Kobold2D is using on an entirely different development philosophy and direction that are essentially incompatible with the official Cocos2D.

Kobold2D exists because Cocos2D simply won't budge.

Focus on Users

Kobold2D is for its users, first and foremost. The users get what they want. There's no big corporation driving the development direction of Kobold2D, just the users.

Ease of use

Kobold2D will take every step necessary to make your development experience as easy as it can be, with as much help as possible from automated tools and clear separation of user code and libraries.

In addition, Kobold2D provides developers with a clean and simple API. For example, GameKitHelper makes using Game Center straightforward, KKInput gives you all the power of gestures and keyboard input without having to learn Apple's APIs.

And of course every new feature of Kobold2D is documented in great detail with lots of example code.

Popular Frameworks included

Kobold2D makes it easy to use the popular frameworks out of the box, be it Lua, the Cocos2D Extensions, Cocos3D, Chipmunk SpaceManager, and so on.

In a nutshell: Kobold2D is a moving target

Kobold2D is going to see progress every month, and it is going to be progress that matters to you because it is based on your feedback and actual game development requirements.

I leave it up to you to figure out what the development philosophy and direction is for Cocos2D.

Should I try Cocos2D first before switching to Kobold2D?

No.

Kobold2D makes it easier to learn Cocos2D programming! You should learn Cocos2D **with** Kobold2D and skip all the hard parts (like installing templates, setting up a project, etc.). You'll get a lot more templates to get started and 100% working example code to learn from. And so much more.

See also this related question: Do all the Cocos2D tutorials and books work with Kobold2D?

Do I need lots of Cocos2D experience to start with Kobold2D?

No.

The whole purpose of Kobold2D is to make Cocos2D game development easier. This includes the first-user experience. That's why Kobold2D has an installer, project starter and upgrade tools, the entire API References both online and offline, and most of all: many example projects to get started with!

Do all the Cocos2D tutorials and books work with Kobold2D?

Yes, followed by a tiny "but ..."

As far as adding new classes and resources to your project is concerned, you can apply everything you learn from tutorials and books directly to Kobold2D projects.

The "but" part is this: rarely you'll be instructed to make changes to the way Cocos2D is set up. You'll notice this when you're instructed to change code in the app delegate or root view controller classes. In Kobold2D, all those startup settings and function calls have been moved into the `config.lua` script for convenience.

You can either add the code in questions to the app delegate's `-(void) initializationComplete {...}` method, or simply apply the changes to the appropriately named parameters in the `config.lua` file.

Cocos2D version mismatch
If a particular book or tutorial discusses — for example — Cocos2D `v0.99` but the current version is `v1.0` you'll very likely encounter compilation errors simply due to changes in the Cocos2D API (eg renamed classes, removed functionality, etc.). This is neither the fault of Kobold2D nor the fault of the tutorial or book author.

When was Kobold2D created?

Kobold2D was first published in August 2011. Some of the concepts implemented in Kobold2D date back to early 2010.

What does Kobold2D cost?

Nothing.

Kobold2D is free and open source. It's released under the MIT License.

We do sell commercial add-on products and affiliate products to support the development of Kobold2D.

Can I donate to Kobold2D?

No.

First of all: donations bring in very little money. It's like trying to convince a software pirate to buy the software he just downloaded for free. Very, very few do it. Plus everyone thinks there are enough others doing it. So, there you go.

Commercial products are much more effective to cover costs and to subsidize the development of free products. And the users actually gets something valuable in return. Win-win.

In addition, it simply wouldn't be right to sell commercial products and then also accepting donations at the same time. At least not without being fully transparent about what the donations are being used for.

There's also the question of fairness: if the donations do cover all of the running costs, how is the extra money used? How should we split donations among contributors and library authors fairly? How do we provide enough transparency about how high the running costs are and how much of it has been covered through donations?

Simply put: we'd rather have you donate to other library authors who need donations.

Why does Kobold2D consume 400 MB disk space?

Currently, the Kobold2D download is about 140 MB and the size of Kobold2D installed on your computer is over 400 MB. Each new version of Kobold2D will add another 330 MB or more to your hard drive.

Kobold2D includes several libraries and many project templates. But the biggest size factor is the documentation in the `./_Kobold2D_/docs/` folder, around 200 MB. Over 80 MB are consumed by the Xcode docset help files which are copied to the user's Docset folder to make them available in Xcode.

New Kobold2D versions are installed to new folders (eg `Kobold2D-1.0.0`, `Kobold2D-1.0.1` , and so on) to enable migrating projects to newer Kobold2D versions with the help of the Kobold2D Project Upgrade tool.

See also: How can I reduce the installed size of Kobold2D?

How can I reduce the installed size of Kobold2D?

You can safely delete the .docset files in the `./_Kobold2D_/docs/` folder. They're already copied to `~/Library/Developer/Shared/Documentation/DocSets/`. You can also delete the docset files in the latter directory, if you don't use the Xcode help for Kobold2D libraries at all.

If you do not need the (somewhat faster) Kobold2D offline documentation you can safely delete the entire `docs` folder. In particular you might consider doing this with older versions of Kobold2D that you don't actively work with anymore, but want to keep around because a project built with this particular Kobold2D version is already in the App Store. You never know when that project might need a bugfix or compatibility update for a new iOS version.

The Kobold2D Online Documentation allows you to access older versions of the documentation.

Of course, you can also delete old Kobold2D versions entirely once you have migrated all of your projects to newer Kobold2D versions. Do this by deleting the folder with the version number suffix, for example:
`Kobold2D-1.0.0-preview`

How can I install Kobold2D to a custom folder?

You can move the `~/Kobold2D` folder after installation to wherever you want.

There's only one thing to consider: for the Kobold2D Project Upgrader tool to work every versioned Kobold2D folder (eg. `/Kobold2D-1.0`) should be in the same folder, so that the folder structure looks something like this:

```
/MyProjects/KoboldProjects/Kobold2D-1.0
/MyProjects/KoboldProjects/Kobold2D-1.1
/MyProjects/KoboldProjects/Kobold2D-1.2
/MyProjects/KoboldProjects/Kobold2D-1.3
```

Otherwise the Kobold2D Project Upgrader tool won't be able to find upgradeable Kobold2D Projects.

The Kobold2D installer doesn't allow you to install Kobold2D to a different folder. It will always install Kobold2D to `~/Kobold2D`. This is done so that users without a system administrator account can install Kobold2D.

How can I uninstall Kobold2D?

It's simple: delete the folder `~/Kobold2D` or a particular versioned folder, for example:
`~/Kobold2D/Kobold2D-1.0`

This will also delete any Kobold2D projects that you have created.

If you want to uninstall in order to re-install Kobold2D, you can simply run the installer again. It will overwrite any Kobold2D files that already exist but none of the files you created.

Removing the Xcode Docsets

To remove the Kobold2D Xcode docsets locate this folder:

```
~/Library/Developer/Shared/Documentation/Docsets
```

The *Library* folder may be hidden, if that is the case, use the Finder menu:
`Go -> Go To Folder` (Command+Shift+G). Then enter the above path in the "go to folder" text field.

You can savely delete all docsets that begin with *com.kobold2d*.

Kobold2D Xcode FAQ

- Why does Kobold2D build successfully but won't run?
- Why do I get "ld .. failed with exit code 1"?
- Is Kobold2D compatible with Xcode 3?
- Does Kobold2D work with the latest beta SDK?
- How can I create a blank Kobold2D project?
- Can I add the Kobold2D.xcodeproj to my project?
- Can I work in a copy of the Kobold2D Workspace?
- Why does Xcode crash after installing Kobold2D or the Xcode Help files?
- How can I copy the Build Log in Xcode?

Why does Kobold2D build successfully but won't run?

If Kobold2D builds successfully but won't open the iOS Simulator, a Mac OS window, or deploy to the device, you may not have created a Kobold2D project with the Kobold2D Project Starter tool after installation.

Solution #1: Adding a Project

If you open the Kobold2D.xcworkspace, and it only contain a single project named `Kobold2D-Libraries.xcodeproj` (contains Kobold2D's static library projects) then there's nothing that *can* be run.

Creating a Kobold2D Project explains how to create a Kobold2D project that you can run.

Solution #2: Selecting the Project Scheme

If you do have a custom project in the workspace, you may not have the build scheme for that project selected in Xcode. Use the drop-down list on the upper left area of Xcode to select the scheme for your project.

By default Xcode will select the "Box2D-iOS" scheme. Box2D is just a static library supposed to be linked with other code, it can not be run by itself. That's why the "code builds but won't run".

Why do I get "ld .. failed with exit code 1"?

There are several causes for this kind of error. The error you're getting may look something like this:

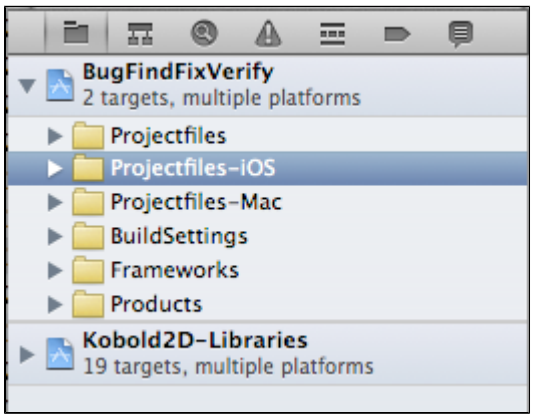
```
ld: file not found:
/depot/_buildoutput/NameOfProject-ebnhlmxjqacipefeghyqtdvdgbvt/Build/Products/Debug-iphonesimulator/libkc
/Developer/Platforms/iPhoneSimulator.platform/Developer/usr/bin/clang failed with exit code 1
```

The solutions are in the order of "most common cause" from top to bottom:

- What it should look like
- Do not open more than one Kobold2D .xcworkspace at the same time
- Open the .xcworkspace, not the .xcodeproj
- Do not create custom Build Configurations
- Archive Builds: make sure scheme does not contain spaces
- Kobold2D-Libraries.xcodeproj is missing
- Kobold2D-Libraries.xcodeproj is contained within your project
- Change the Xcode DerivedData folder location

What it should look like

Before you begin troubleshooting, this is how a healthy Kobold2D workspace looks like. It contains one or more of your own projects, plus the Kobold2D-Libraries project at the same level:



A correctly loaded Kobold2D workspace has the Kobold2D-Libraries project in it.

Do not open more than one Kobold2D .xcworkspace at the same time

Let's say you've created two workspaces via the Kobold2D Project Starter tool and named them MyFirst.xcworkspace and MySecond.xcworkspace. Both workspaces will include a reference to the Kobold2D-Libraries.xcodeproj. Now, if you open MyFirst.xcworkspace everything will be fine. But if you then open MySecond.xcworkspace before closing the MyFirst.xcworkspace Xcode will unfortunately not load the Kobold2D-Libraries.xcodeproj.

You'll find that the Kobold2D-Libraries.xcodeproj is displayed merely as a text entry and can not be expanded:

Kobold2D-Libraries.xcodeproj is displayed as text entry because it is in use by another .xcworkspace - close all workspaces, re-open just the one you need.

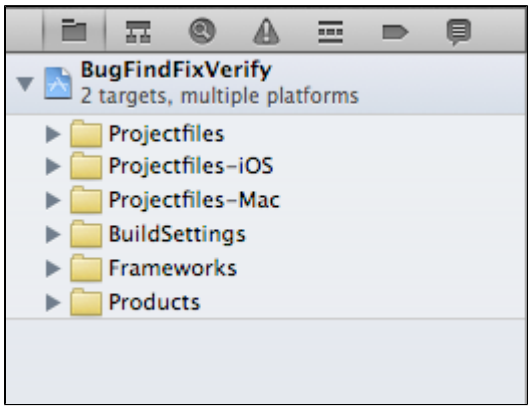
When you build any project in MySecond.xcworkspace you'll quickly receive the "file not found … failed with exit code 1" error.

To fix the problem, close both workspaces, then open only MySecond.xcworkspace. See also this Stackoverflow post which indicates that this is an Xcode bug.

You can make one or more copies of the entire **Kobold2D-x.x.x** folder to be able to open one .xcworkspace from each folder at the same time.

Open the .xcworkspace, not the .xcodeproj

You most likely opened the .xcodeproj file. Kobold2D projects only work if you open the accompanying .xcworkspace file that contains (references) the .xcodeproj of your project.



There's no **Kobold2D-Libraries.xcodeproj** here because the .xcodeproj was opened rather than the .xcworkspace.

A sure way of knowing whether this is your problem is to have a look at the Project Navigator pane (View -> Navigators -> Show Project Navigator — hotkey: Command + 1). There must be a **Kobold2D-Libraries** project with a blue icon at the root level. If it doesn't exist, you've opened the .xcodeproj file instead of the .xcworkspace file.

If you did open the .xcworkspace file but the **Kobold2D-Libraries** project isn't there, you may have accidentally removed the project. You can either add the Kobold2D-Libraries.xcodeproj back in at the root level, or create a new workspace with the Project Starter tool, then drag and drop your project into that workspace at the root level.

Do not create custom Build Configurations

Unfortunately, at the moment Kobold2D will fail to link with its libraries if you copy one either of the two existing **Debug** or **Release** build configurations. Commonly developers use this to create "AdHoc" or "Distribution" configurations.

The good news is: there are no additional build configurations necessary to create Ad-Hoc or App Store distribution builds with Xcode 4. Instead, simply run a Product -> Archive build. When finished, the Organizer window will appear. From there you can choose to share (Ad-Hoc) or submit (App Store) your built app.

I made an inquiry on stackoverflow regarding this issue which also explains why the issue occurs in the first place: the linker is not looking in the correct folder for the static libraries.

Archive Builds: make sure scheme does not contain spaces

If your project only fails when making a Product -> Archive build, then check your project's scheme name. If the scheme's name contains a space like in "My Project" then Xcode will fail to create an Archive build.

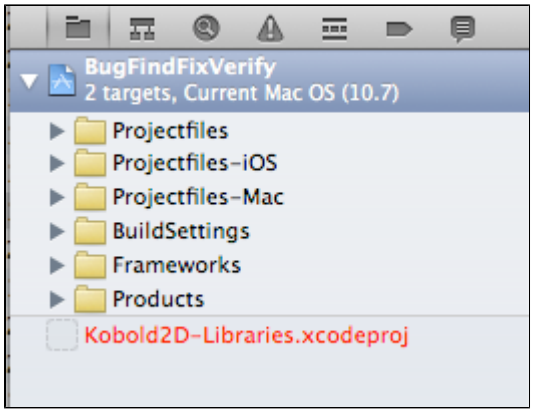
This happens when you rename your scheme, or simply when you duplicate a scheme and Xcode prepends "Copy of ".

Since other characters may also cause the Archive Build to fail you should stick to the "save" characters for scheme names: letters, digits, dashes (-) and underscores (_). Brackets () also work fine.

For example this scheme name will work: "My-Project_(RELEASE)" whereas this won't: "My-Project (RELEASE)" because of the space character. Almost unbelievable, but true and confirmed by other Xcode users.

Kobold2D-Libraries.xcodeproj is missing

If the Kobold2D-Libraries.xcodeproj is displayed in red like in the screenshot below, the project couldn't be found.



Kobold2D-Libraries.xcodeproj is displayed in red, indicating that the project couldn't be found at the path `./_Kobold2D_/Kobold2D-Libraries.xcodeproj`

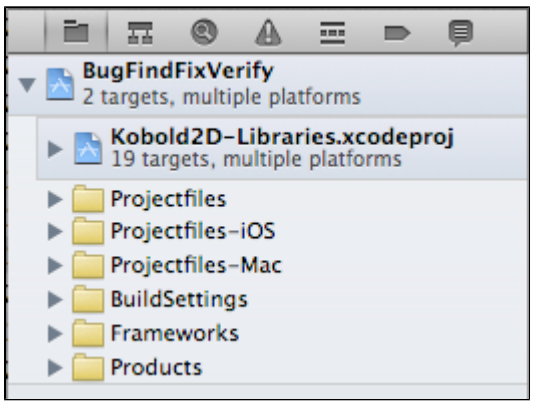
This can have many causes, from accidentally deleted or renamed files or folders down to a corruption of the Kobold2D installation. The preferred way to fix this is to:

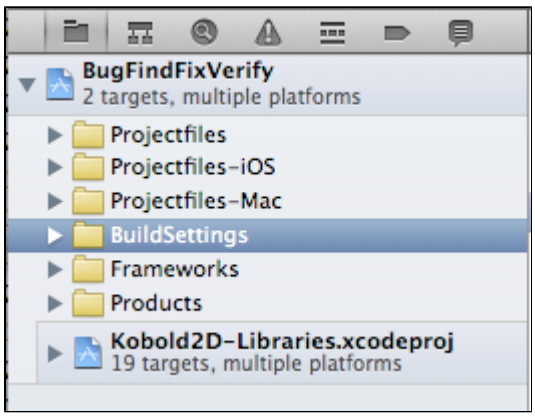
1. re-install Kobold2D
2. re-create a .xcodeworkspace with Kobold2D Project Starter tool
3. (if necessary) re-add your .xcodeproj to the newly created .xcodeworkspace

If your .xcodeworkspace is merely missing the Kobold2D-Libraries.xcodeproj then you can usually skip re-installing Kobold2D (1) and start with re-creating a new .xcodeworkspace (2).

Kobold2D-Libraries.xcodeproj is contained within your project

This should not normally happen unless you manually drag & drop the projects into a .xcworkspace. This is what it might look like:





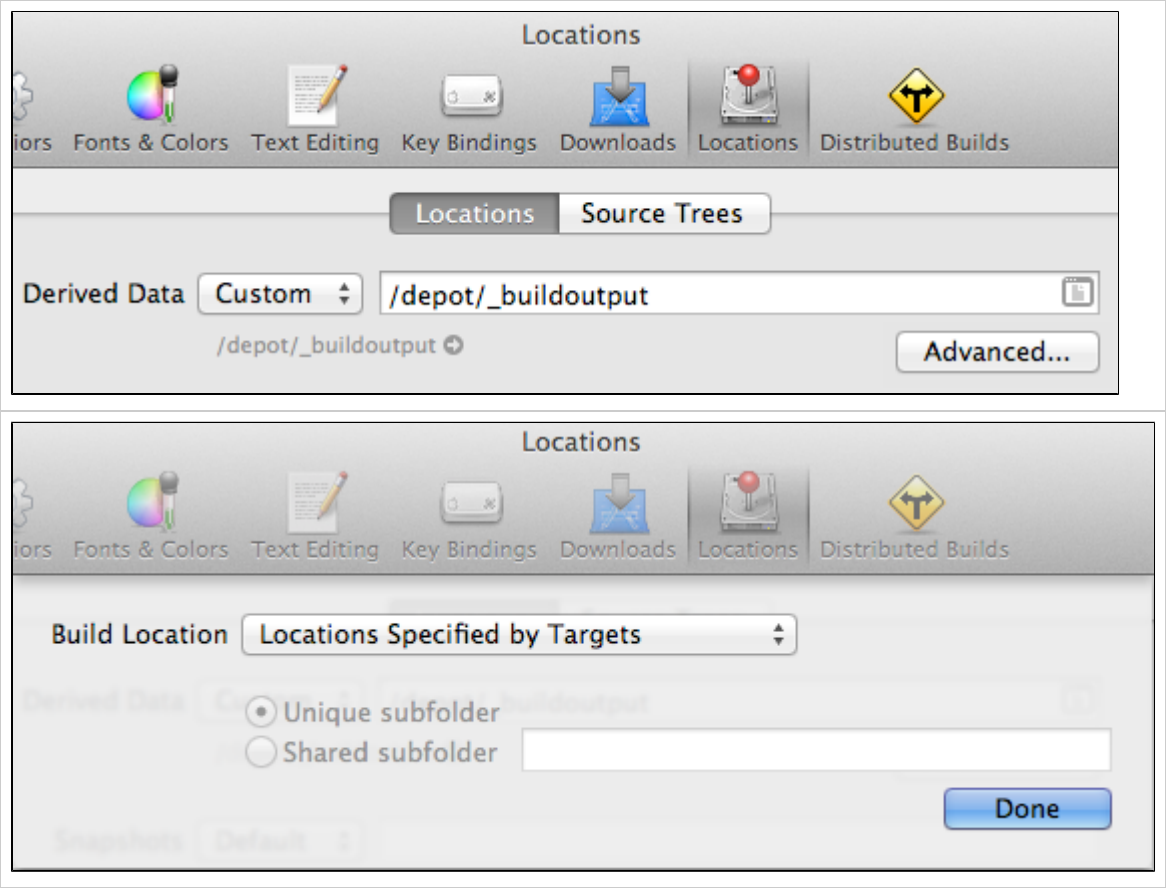
If the Kobold2D-Libraries project appears as a sub-project under one of your projects, you should remove the reference (don't delete the project and its files!) and re-add the Kobold2D-Libraries project to the workspace by dragging and dropping it. Then close and reopen the workspace.

If you're having difficulty with this, simply create a new workspace with the Kobold2D Project Starter tool and drag your .xcodeproj into that workspace. Make sure you drop the project on the far left side of the Project Navigator pane, otherwise it will be added as a sub-project to one of the existing projects, and you would face the exact same problem again.

Change the Xcode DerivedData folder location

Several users reported a persistent "ld .. failed with exit code 1" error on various systems. So far it is unclear what is causing it.

In this thread one user reported that it helped to change the DerivedData location. Under **Xcode Preferences -> Locations -> Advanced** the Build Location was changed to **Location Specified by Targets**. After making this change the builds succeeded.



It can't hurt to test changing the DerivedData path to a location where you are guaranteed to have write permissions and enough free disk space (or quota). The size of the DerivedData folder can easily grow to several gigabytes! Thus the failure could also be related to running out of disk space or not having write permissions.

It was also suggested that changing the Home folder location to a different drive, respectively installing Xcode to a different drive other than the system drive might be causing issues. So far this has not been confirmed.

Is Kobold2D compatible with Xcode 3?

No.

Kobold2D uses a workspace to combine multiple projects into a single workspace. This, among other improvements that Kobold2D makes use of, is only available in Xcode 4.

> You can have both Xcode 3 and Xcode 4 installed on your system. Just install each SDK to a different folder on your hard drive. Instead of the default `/Developer` you could install to `/Developer-XC3` and `/Developer-XC4`.

Does Kobold2D work with the latest beta SDK?

Maybe, maybe not.

> **No beta SDK Support!**
> Kobold2D generally doesn't support Apple iOS SDK beta or Mac OS X SDK beta builds. Any incompatibilities will not get looked at until Apple has released a GM Seed or final version of the latest SDK.

Kobold2D is somewhat at the mercy of other libraries. For example, cocos2d-iphone also fixes incompatibilities only after the new SDK has been officially released.

General Tips About Apple's Beta SDKs

Do not talk about beta SDKs!
Rule #1: Do not talk about Apple beta SDKs!
Rule #2: Do not talk about Apple beta SDKs!

There's a reason for that: beta SDKs are under NDA (non-disclosure). You are not allowed to publicly talk about beta SDKs. Once you allow users to discuss issues that appear due to Apple's beta SDKs the likelihood that someone mentions something that is under NDA is high. Similarly support is hard and sometimes impossible if you can't mention the very thing that you need to talk about to explain a possible solution.

Always install beta SDKs separately!
If you have to work with a beta SDK, make sure that you install it to a different directory and not `/Developer`. You might want to suffix the `/Developer` folder with the version number of the beta SDK, for example `/Developer51` if you're installing the preview version of iOS SDK 5.1.

You can then switch between the latest official SDK and the beta SDK by running the tools (eg Xcode) from the corresponding folder.

Do you really need the beta SDK?
Unless you **must** develop at the bleeding edge, you should prefer to wait for the public release of the new SDK. You generally don't get support for beta SDKs anywhere but the closed Apple Developer forums, and help for particular libraries and compatibility issues with them is minimal at best.

Only consider installing a beta SDK if:

- you have to use one of the new features in your App.
- you have to test your App very early for compatibility issues, for example because you want your app to be among the first to support a new Apple device.
- you are generally curious and know the risks, know how to install the beta SDK to a separate folder, know how to distinguish between official and beta SDK tools such as Xcode, won't ask questions about the beta version in public.

In any other case it is better to wait for the new SDK to be officially released.

How can I create a blank Kobold2D project?

Use the **Kobold2D Project Starter.app** as described in Creating a Kobold2D Project and then modify the newly created project.

Only Kobold2D Project Starter.app created project are supported!
While you can create and add any project to the Kobold2D workspace, such projects will not benefit from the Kobold2D features. The same goes for existing non-Kobold2D projects being added to the Kobold2D workspace.

The process of correctly creating a working Kobold2D project is very involved and not documented. It is a lot easier to simply modify one of the provided template projects. If you want to migrate an existing project to Kobold2D, you would also start by first creating a new Kobold2D project.

Can I add the Kobold2D.xcodeproj to my project?

That's not supported.

You can try and add the Kobold2D.xcodeproj to an existing project, but it won't magically turn your existing project into a Kobold2D-enabled project. More likely you'll get compile errors that we'll be unable to help you with.

Likewise, just adding your project to the Kobold2D.xcworkspace will not make your project a Kobold2D-enabled project.

Instead, this question ultimately seems to be about Migrating a Cocos2D Project.

Can I work in a copy of the Kobold2D Workspace?

Yes, as long as the copied xcworkspace resides in the same folder as the `Kobold2D.xcworkspace`.

The Kobold2D Project Starter tool allows you to create new projects in a custom workspace simple by choosing an existing workspace, or entering the name of a new workspace.

Why does Xcode crash after installing Kobold2D or the Xcode Help files?

> **Close Xcode during installation**
> It is recommended to quit Xcode before installing Kobold2D.

Xcode may crash if it is running while performing one of these actions:

- installing a new version of Kobold2D
- re-installing the same version of Kobold2D
- running the `/_Kobold2D_/docs/install-all-docsets.sh` script

The cause of the crash is that Xcode Help does not like existing and open docsets (Xcode help files) to be removed or replaced. This is what happens in the above cases and could cause Xcode to report an error, after which you are prompted to either "Ignore" the error (which may not work) or "Crash" Xcode.

How can I copy the Build Log in Xcode?

For reporting compile-time issues with Kobold2D you may be asked to post the full build log. Here's how you can copy the build log in Xcode 4:

1. In Xcode, open the **View** menu and choose **Navigators -> Log Navigator** (hotkey: Command + 7).
2. Select the topmost item in the Log Navigator list that starts with "Build".
3. The build log is to the right of the Log Navigator. Click on any log entry just to change the focus to the build log panel.
4. Then choose **Edit -> Select All** (hotkey: Command + A) followed by **Edit -> Copy** (hotkey: Command + C). This copies the entire build log as text to the clipboard
5. You can then paste the build log anywhere.

> The log can be quite long, if it is too long to post in the forum, save the log as text (eg with TextEdit.app) and attach the txt file to the post or email it.

Kobold2D Technical FAQ

- Can Kobold2D Projects be located in a custom folder?
- Where does the 'Network Connections' dialog come from?
- How to disable iSimulate?
- Why is the AppDelegate class empty?
- Are Kobold2D apps larger than pure cocos2d-iphone apps?
- How to fix compile errors with CocosBuilder?
- Why does Kobold2D compile libraries I don't use?

Can Kobold2D Projects be located in a custom folder?

Theoretically yes, but it's unsupported and strongly discouraged.

Every new project you create with the Kobold2D Project Starter app will reside in the versioned Kobold2D folder (eg `/Kobold2D-1.0`). The project will either be added to the Kobold2D.xcworkspace or reside in its own workspace. Neither the workspace nor the project should be moved to a different folder, it will stop working if you do so.

Most importantly, if you move your workspace and project outside the versioned Kobold2D folder, the Kobold2D Project Upgrader tool will be unable to upgrade that project to a newer Kobold2D version. Other tools may also stop working.

Possible Solution for Power Users (Unsupported)

> **Voiding Warranty**
> The following steps are for those who feel confident in changing the Kobold2D project structure. We strongly recommend that you do not do this, and we will not give support for the following modifications.

Please ask yourself what reasons you have for changing the Kobold2D project structure. We programmers often like to have things our own way and customize our work environment because it just feels right. But often we do so without even trying to solve very particular issues and without consideration for all the potential drawbacks of that customization.

The "still ok" solution to custom folders

Kobold2D workspaces rely on the `Kobold2D-Libraries.xcodeproj` to be in the relative path `/__Kobold2D__/Kobold2D-Libraries.xcodeproj`. If you move your workspace to a custom folder it obviously won't be able to find the `Kobold2D-Libraries.xcodeproj` anymore. While you can simply change the path to the `Kobold2D-Libraries.xcodeproj` you should ask yourself if this is really what you want.

The much better alternative would simply be to copy the `__Kobold2D__` folder along with your project and workspace, and so your workspace continues to work and your project is tied to a very particular version of Kobold2D.

If you keep all of your custom folders at the same level even the Kobold2D Project Upgrader tool would still work.

The "inconvenient but workable" solution to custom folders

If you still decide to keep your workspace and project in a different folder from the `__Kobold2D__` folder, be aware of the following issues:

For one, you'll have to dig deep in your workspace to find out with which `Kobold2D-Library.xcodeproj` your workspace is currently linked with. It could be the one in the `/Kobold2D-1.0` folder or it could be the one in the `/Kobold2D-1.1` folder, or some other version.

Without actually looking it up in the workspace you won't know. And you will have to maintain that connection manually in your workspace. You should also avoid renaming or moving the Kobold2D versioned folders because you'll never know which of your custom-located workspaces is linked with that particular `Kobold2D-Library.xcodeproj`.

It's an inconvenient but workable solution.

The "thing that should not be" ~~solution~~ recipe for disaster to custom folders

If your intention is to keep only a single version of Kobold2D and its libraries in a commonly named folder (eg simply `/Kobold2D`) then every time you upgrade that Kobold2D version, all of your custom workspaces and projects will be using the latest Kobold2D version automatically. That may seem like a good idea at first, but it's really not. It's the worst possible setup you can have.

Consider this: You've built an app with Kobold2D-1.0 that's been in the App Store for 6 months. The project remained untouched ever since. Now you have to update that App due to an iOS update and an incompatibility caused by it. You open the project only to find out that it doesn't build anymore because in the meantime you've upgraded Kobold2D to v2.0. Instead of a quick fix you'll be forced to adapt your entire project to a new API and re-test everything.

Now you'd think you could just downgrade the Kobold2D version temporarily. But alas, you don't even know for sure which Kobold2D version your project was built with. It could have been v1.0 but it could have also been v1.1 or v1.2.

If you work with source control and label your releases, including all referenced libraries, this won't be that much of a problem for you. But it's still inconvenient and there's the potential for human error (forgot to label a release) that may reveal itself only months later.

Where does the 'Network Connections' dialog come from?

When running iOS Simulator builds of a project, you'll see this network connections dialog pop up:

Do you want the application "_Doodle-Drop-Template_-iOS" to accept incoming network connections?
Clicking Deny may limit the application's behavior. This setting can be changed in the Firewall pane of Security preferences.
Deny
Allow

This is nothing to be alarmed of. It is caused by the iSimulate library that allows you to control the action in the Simulator via a Wifi connection from your iOS Device with the iSimulate app installed.

If you don't want or need iSimulate and/or want to get rid of the popup dialog, please read How to disable iSimulate?

How to disable iSimulate?

You can (temporarily) disable iSimulate by opening your project's `BuildSettings-iOS.xcconfig` file located in the BuildSettings group. Locate the **FORCE_LOAD_ISIMULATE** line and simply comment it out to disable iSimulate for the time being:

```
// Comment this line if you don't want or need iSimulate.
// Doing so will also remove the "incoming network connection"
// warning dialog in Simulator builds.
FORCE_LOAD_ISIMULATE = -force_load $(KKLIBROOT)/iSimulateSDK/libisimulate-4.x-opengl.a
```

iSimulate is only linked in iOS Simulator builds of your app:

```
OTHER_LDFLAGS[sdk=iphonesimulator*][arch=*] = $(OTHER_LDFLAGS) $(FORCE_LOAD_ISIMULATE)
```

It is not necessary to remove iSimulate from distribution builds.

Why is the AppDelegate class empty?

The `AppDelegate` class inherits from `KKAppDelegate` which does most of the work for you, including the configuration of Cocos2D and the startup process via the `config.lua` file.

You can still add any `NSApplicationDelegateProtocol` method to the `AppDelegate` class as you see fit. But if you do, you should be sure to call the `super` implementation of that method to allow `KKAppDelegate` to do its job.

Cocos2d-iphone has often caused headaches for developers who updated to a new cocos2d-iphone version but weren't aware that you also needed to make changes to the Cocos2D startup code in your project's app delegate class. Kobold2D avoids this problem by keeping the startup code internal in the `KKAppDelegate` class.

Are Kobold2D apps larger than pure cocos2d-iphone apps?

Typically: no.

Tests with release builds of Kobold2D template project and comparing the results with the cocos2d-iphone project templates showed that the resulting Kobold2D apps are often slightly smaller (several KB) despite the fact that Kobold2D has the entire Lua library built into the app.

Kobold2D does compile more source code but the resulting app size in release builds is typically smaller than projects created with the cocos2d-iphone Xcode templates.

How can Kobold2D apps be smaller than cocos2d-iphone apps?

Kobold2D links the third party code into your app as static libraries. By default, all libraries are added to the **Link Binary with Libraries** build

phase. But here comes the clever part: the linker actually knows which libraries are needed and which aren't. Any libraries that you don't use in your app will not be linked with your app and **don't affect your app size**.

> The benefit for you is convenience without any drawbacks: If you want to start using a library, it's as simple as adding the correct header file(s) to your source code.

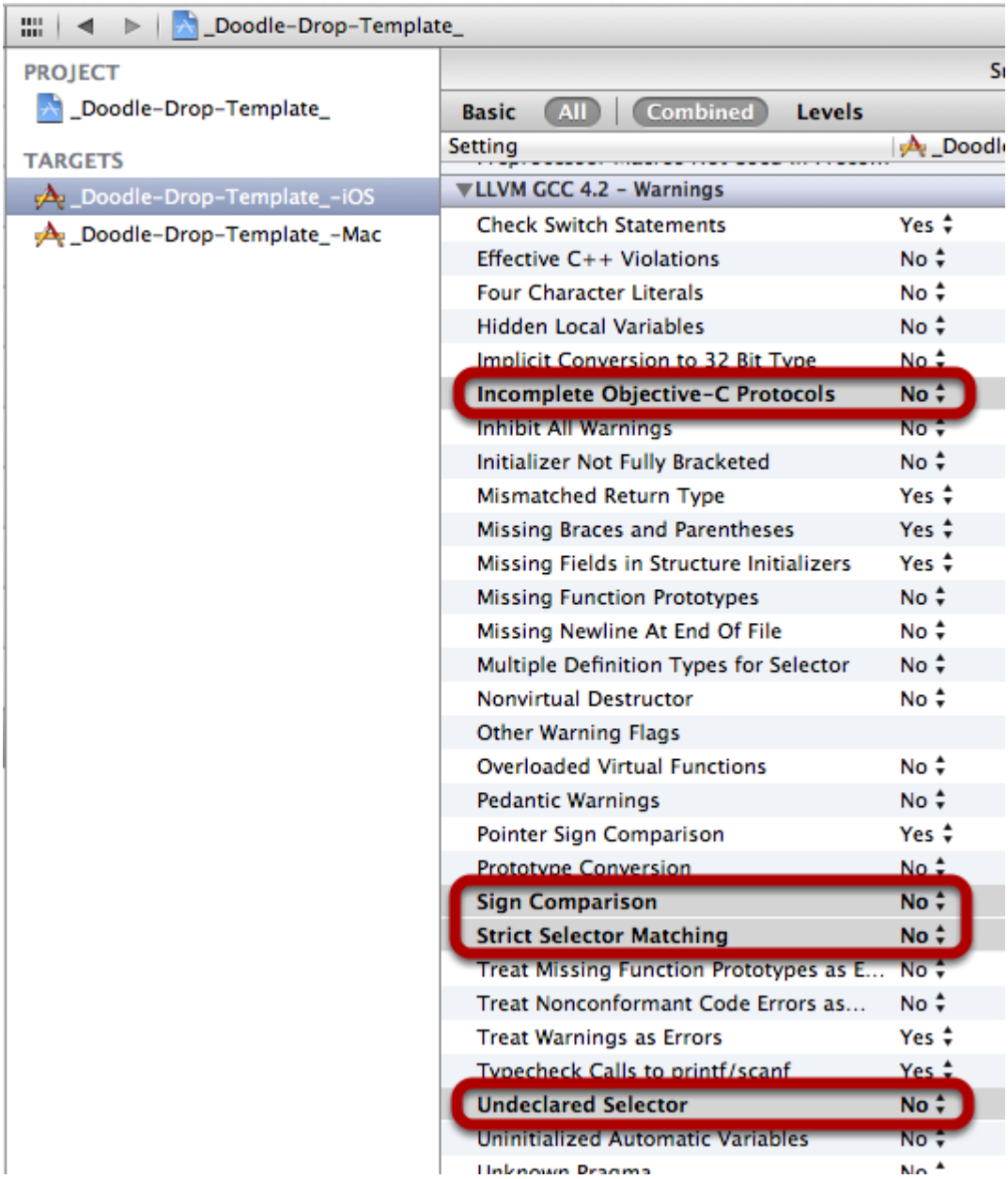
Next up is dead code stripping. The linker is an intelligent beast. Even if you use a library, you'll hardly use all of its functions. Any unused functions are simply removed from the app. There are indications that the linker can do a better job of stripping dead code if that code resides in a static library.

Finally, several optimizations have been applied to the global build settings of Kobold2D projects, including proper debug symbol stripping in release builds and disabling C++ runtime type information and exception handling, that lead to a smaller executable size.

How to fix compile errors with CocosBuilder?

The CocosBuilder editor for Cocos2D comes with a class called `CCBReader`. If you add the class files to your Kobold2D project it will no longer compile without errors because the CCBReader class doesn't adhere to Kobold2D's stricter warning protocol.

The easiest fix is to disable (set to **NO**) the following warnings in your project's target(s) after adding the CCBReader class:



Why does Kobold2D compile libraries I don't use?

Kobold2D provides all third party libraries as preconfigured, static libraries. By default, all libraries **link to** a Kobold2D project. This causes all the library code to be compiled, whether you use it in your project or not. This is **done only once** for each build configuration.

> **Don't worry!**
> If you concerned about your final app's size, please read this FAQ:
> Are Kobold2D apps larger than pure cocos2d-iphone apps?

Note that the compilation of the entire Kobold2D code base happens only the first time you compile a new version of Kobold2D, or after switching build configuration (Debug vs Release, iOS vs Mac, iPhone vs iPad vs Simulator), or after a "clean" build.

If you build other projects within the same Kobold2D version, or create a new Kobold2D project, only the project's code will be built since the static library code is already built and available for other projects.

Put simply: from the second build of the same build configuration onwards you're actually saving time!

Kobold2D Cocos2D FAQ

- Is Kobold2D compatible with cocos2d-iphone?
- Is Kobold2D updated when cocos2d-iphone is?
- Will installing Kobold2D affect my Cocos2D Projects?

Is Kobold2D compatible with cocos2d-iphone?

Yes.

If you create an empty Kobold2D project and add your existing cocos2d project's source code and resources to it, and assuming the cocos2d-iphone versions are identical, there's no reason why your code should not continue to work.

Is Kobold2D updated when cocos2d-iphone is?

Yes.

As soon as we hear about a new stable release of cocos2d-iphone, we will integrate and test it with Kobold2D. If everything works fine, a new release of Kobold2D will be released within days (hopefully just hours) after the new cocos2d-iphone release.

> If any of the other libraries are updated (eg. cocos3d), we normally aim to include the update in the next scheduled Kobold2D update and won't make an immediate release.

Will installing Kobold2D affect my Cocos2D Projects?

No, not in any way.

Your Cocos2D Projects remain completely separate from the Kobold2D installation. Kobold2D doesn't overwrite or otherwise modify Cocos2D or its Xcode templates.

But maybe you are interested in Migrating a Cocos2D Project? You can do so safely. If you follow the migration guide your Cocos2D project also remains unchanged.

Kobold2D Wax & Lua FAQ

- What's the difference between Wax and Corona SDK?
- Can I write game logic in Lua with Kobold2D?

What's the difference between Wax and Corona SDK?

About Corona SDK

The Corona SDK exposes a well designed API in Lua for app and game development. This API is naturally different from the iOS SDK since Corona is also a cross-platform game engine, exposing the same API for all platforms.

For example, when you receive an accelerometer event in Corona you get an event object with these properties. The acceleration parameters are already split into "instant acceleration" and "gravity acceleration", something that iOS developers have to do manually. At the same time, the event also reports whether it was a shake event. On iOS this is an entirely different API from the accelerometer events.

In one sentence, Corona exposes the functionality that the devices offer in a uniform way. Corona also condenses information into fewer properties and methods. You deal with fewer functions which makes it great for rapid prototyping. But sometimes you'll have fewer functionality, and in particular you can't extend Corona with your own Objective-C, C or C++ code since the backend engine is proprietary. Overall, this makes Corona SDK very easy to learn but because it's not extensible and can impose restrictions, it's not for everyone.

About Wax

The iPhone Wax library is a dynamic, automatic binding of the Lua language to Objective-C. Wax provides a translation layer that exposes regular Objective-C code to Lua via Objective-C runtime functions.

Since the binding happens at runtime, it's a **dynamic binding** — in other words there is no code that says that function X should be exposed to Lua in a particular way. Instead, there's code that creates the Lua methods and properties by analyzing the existing Objective-C code at runtime, and applying generic conversion mechanisms to translate between the two languages. The **binding is automatic** because that translation is transparent to the user.

Wax in essence allows you to write Objective-C code in the scripting language Lua. This is nice, in particular automatic memory management should be mentioned as a positive feature of Wax. But other than that it's really hard to justify the use of Wax if you can write the same code in Objective-C. Here are the major drawbacks of writing (scripting) apps with Wax:

1. You end up writing almost the exact same code as in Objective-C. It may be a little less code but it's using a weird and uncommon syntax.
2. The dynamic nature of Wax' Lua binding causes a significant overhead. So much in fact that actual gameplay scripting for realtime (60 fps) games will result in disappointing performance — unless the game is rather simple and/or you target only newer devices (4th generation onward, and iPads).
3. You can not properly debug Lua scripts. You can't set breakpoints, you can't single-step through script code, you can't inspect variables.
4. Lua is a dynamically typed language. That means the compiler won't check Lua scripts for errors. Any syntax errors in your script will only surface while your app is already running.
5. Lua support in Xcode is non-existent. There's no syntax highlighting, no auto-completion, no refactoring, no quick help for Lua script files.
6. You lose named parameters in Lua code. Lua functions are non-descript like C functions: `someFunction("myName", 1, true, "triggerMe", false, 100, 0.4)`

In particular not being able to debug your Wax scripts (#3) and not having compile-time syntax checks (#4) will completely offset any time-savings you might get from writing your app entirely in Wax.

Wax doesn't play into Lua's strengths, which is that it's usually used to expose a simplified, domain-specific API. Such an API would be less error-prone, is easier to understand and debug. Instead, Wax exposes the full complexity of Objective-C and applies a weird syntax, removes debugging, disables syntax checking and reduces code editing comfort. Not on purpose, mind you, but in effect that's what you get.

While Corona shares the issues #4 to #6 with Wax, it's performance and code design is well above that what Wax is able to provide. Corona also comes with a Lua debugger, albeit a command-driven one (eg. like Terminal).

The Solution: A Native Lua Scripting Language

Lua is not without its advantages. It's an excellent language to script games in. But this is only true if the purpose of the language has been determined and an appropriate API has been designed and implemented, exposed to Lua via regular, non-dynamic Lua bindings for optimal performance. That requires a lot of work.

You can bind the exposed API to Lua either by manually writing C functions that manipulate the Lua stack, or via binding libraries such as tolua. Such an approach can have one of these goals:

1. A general purpose game engine API for writing apps in Lua. (Corona SDK)
2. A domain-specific scripting API to manipulate existing game objects at runtime for a specific purpose. This is often tailored to work only with a particular (type of) game. (World of Warcraft UI Scripting)
3. A "load-time" API to configure the app and its objects. Reduces the amount of boilerplate code needed for setting up a scene. Can be used by add-ons and tools as a user-editable data format for scenes, levels, etc. (Kobold2D)
4. Use Number 3 to configure a domain-specific, runtime state machine. The statemachine is implemented in a high-level language (Objective-C, C, C++) for best performance. The high-level API is exposed through Lua to provide a user-friendly API to designers. The Lua scripts are run once to initialize the statemachine. (Battleforge, Spellforce)

Number 3 is already implemented by Kobold2D. Number 2 and/or 4 could be implemented in the future, provided that there's a common module that the scripts are supposed to control. For example, this could be a scriptable menu system so that menu screens can be designed and programmed entirely in Lua. This is **not** a promise for a feature, just something that *could* be done.

Number 4 is quite powerful, but new developers repeatedly find it confusing to write scripts that are executed exactly once (eg each time the level is restarted) because the scripting API implies a runtime nature that it doesn't have — it's only a setup/init script that's translated to a runtime state-machine.

Number 1 is not compelling — for one users can simply use the Corona SDK if they want to write everything in Lua, and secondly some of the drawbacks for Wax still apply: no debugging, no compile-time error checking, no Xcode support.

In addition, implementing Number 1 on top of Cocos2D means that there will have to be some amount of hacks, compromises and extension code to keep the Lua API clean and simple while making it work with the already existing Cocos2D API. It would make more sense to write an entirely new game engine with a Lua interface, and also make that cross-platform. But wait, that's what the Corona SDK is! You also get such a Lua interface with Cocos2D-X.

Can I write game logic in Lua with Kobold2D?

In theory yes. But no.

Kobold2D does not provide a Lua scripting interface for game engine classes, eg you can't script game logic in Lua. What you can do is to edit your game's parameters in Lua, then read them in with just one line of code.

Kobold2D relies on Wax which uses the Objective-C runtime to create dynamic bindings to Lua. So you can write Objective-C code in Lua. However, there's really little benefit to that. You lose the ability to debug your code and you'll have to handle awkward syntax that's almost like Objective-C but not quite. The only benefit is automatic memory management, something that Automatic Reference Counting (ARC) will address with iOS 5 anyway.

The Wax binding code through Objective-C runtime features also adds a significant overhead so that scripting games with Wax will give you bad or terrible performance. To get a good performance you would have to write C functions manipulating the Lua stack directly for each function that you need in your scripting language. This is currently not available in Kobold2D.

I suggest to check out the Corona SDK if you want to write your entire game using Lua with good performance and also be able to deploy to Android devices.

Kobold2D Audio FAQ

- Which audio engine is better - CocosDenshion or ObjectAL?

Which audio engine is better - CocosDenshion or ObjectAL?

That's subjective. For the most part.

CocosDenshion works on both iOS and Mac OS X. So if you do develop for both platforms you should use CocosDenshion.

If you are developing only for iOS, ObjectAL is a serious contender for two reasons: the API is very well designed, and it's well documented.

Audio Engine Comparison Chart

	CocosDenshion	ObjectAL
iOS	✓	✓
Mac OS X	✓	✗
"Simple" Audio Engine	✓	✓
Documentation	CocosDenshion Cookbook CocosDenshion FAQ	ObjectAL Reference Documentation

Kobold2D Physics FAQ

- Which Physics Engine is the best?

Which Physics Engine is the best?

Allow me ...

There's no "The Best"

There's only the physics engine that **better suits you**. Just like music.

	Box2D	Chipmunk
Language	C++	C
Documentation	Manual & API Reference	Manual & API Reference
Arithmetics	`vec = vec1 + vec2 - vec3;`	`vec = ccpSub(ccpAdd(vec1, vec2), vec3);`
Optimized for	Objects of varying sizes	Objects of similar sizes
Special Features	Bullets (aka continuous collision integration aka swept collisions) Operator overloading	-
Requirements	Must use .mm extension Objects should be sized between 0.1 to 10 units	-

Box2D now has a Objective-C wrapper dubbed Boxjective2D available in KoboldTouch. There's also Objective Chipmunk, a commercial Objective-C wrapper for Chipmunk by the Chipmunk developer.

A free and popular Objective-C wrapper alternative is the Chipmunk SpaceManager (included in Kobold2D), but it is tied into Cocos2D and whenever the Cocos2D API changes, it requires the SpaceManager code to be updated accordingly. In the past it sometimes took weeks or months before the official SpaceManager code was updated. The SpaceManager is also not well documented, providing only an introduction and the API Reference.

There's no similarly popular Objective-C wrapper for Box2D. The CCBox2D project seems promising but like SpaceManager it's not a generic wrapper but depends on Cocos2D.

Which one is easiest to learn?

Again, this depends on you. 50% of developers who swear that Box2D is easier to learn than Chipmunk fail to understand why the other 50% find Chipmunk easier to learn than Box2D.

The % are made up, but you get the point.

Things to consider before deciding on a physics engine

Some people would think the choice should mainly be based on performance. Wrong. Usability, familiarity with the programming language and documentation are far more important in choosing any library almost always. Even more so if you have little to no experience with physics engines.

There are some special cases where Chipmunk may be faster, other cases where Box2D may be faster. If you know what cases that might be you should perform some simple tests and measure the performance. Note that Chipmunk SpaceManager will always be slower than pure Chipmunk, but it makes up for being easier on you if you already know and like Objective-C.

If you know what kind of joints you are likely going to need, check the documentation for each physic engine. Chipmunk may have some joint types that Box2D doesn't have, and vice versa. This could easily be a killer argument for or against a certain physics engine.

If you are going to have very fast moving physics objects, eg "Bullets", consider using Box2D as it can do swept collisions aka continuous collision integration to prevent fast-moving objects from deeply penetrating or even tunneling through other objects.

If you find one physics engine more "accurate" or "stable" than the other, eg. stacking objects wiggle more or less, know that some of these issues can be resolved by changing the number of iterations the physics engine performs. This also affects performance. It is not unusual for iOS games to reduce the number of iterations to 1 or 2 for performance reasons. You might want to try that and see which physics engine requires how many iterations to simulate your game with the needed accuracy before introducing issues like "restless" objects.

If you prefer object-oriented development, have a history with C++ or prefer speaking variable names, you should choose Box2D. But in Kobold2D projects you will have to change the .m file extension to .mm and stick with that to compile the code as Objective-C++ code. Failure to do so even once will greet you with dozens, if not hundreds of compile errors. Don't Panic!

If you are a purist and prefer C and don't mind dealing with variable names like `e`, `f` and `m` then try Chipmunk or Chipmunk SpaceManager. Chipmunk code is also not as verbose as Box2D code due to its naming scheme, but then again it doesn't provide overloaded operators so all arithmetics are done more verbose and less intuitive via function calls: `ccpVect vec = ccpSub(ccpAdd(vec1, vec2), vec3)`.

Kobold2D Cocos3D FAQ

- Is there a Mac OS version of Cocos3D?

Is there a Mac OS version of Cocos3D?

No.

It's listed on the development roadmap, but very far down.

Kobold2D Release Notes

Kobold2D v2.1.0 Release Notes

The Most Important Changes & Additions

- compatible with Xcode 4.6 (fixed FIX_CATEGORY_BUG issue and others)
- improved App delegate to properly support iOS 6 autorotation

Minor Improvements & Bug Fixes

- enabled armv7s architecture (iPhone 5, iPod touch 5, iPad 3/4) in all targets

Libraries updated

- updated: cocos2d-iphone 2.1 rc0a
- updated: cocos2d-iphone-extensions 0.21 (latest from github)
- updated: Chipmunk 6.1.2
- updated: Chipmunk Spacemanager 0.2.01
- fixed: SneakyInput compiles with 2.1
- fixed: Chipmunk Spacemanager compiles with 2.1 (setOpacity & setColor without function)
- removed: AdMob SDK

Known Issues

- Chipmunk SpaceManager is not fully compatible with cocos2d 2.1 (color and opacity changes may not be applied to sprites etc handled by SpaceManager)

All Release Notes

- Kobold2D v2.x.x Release Notes
- Kobold2D v2.1.0 Release Notes
- Kobold2D v2.0.4 Release Notes
- Kobold2D v2.0.3 Release Notes
- Kobold2D v2.0.2 Release Notes
- Kobold2D v2.0.1 Release Notes
- Kobold2D v2.0.0 Release Notes
- Kobold2D v1.x.x Release Notes
- Kobold2D v1.1.3 Release Notes
- Kobold2D v1.1.2 Release Notes
- Kobold2D v1.1.1 Release Notes
- Kobold2D v1.1.0 Release Notes
- Kobold2D v1.0.5 Release Notes
- Kobold2D v1.0.4 Release Notes
- Kobold2D v1.0.2 Release Notes
- Kobold2D v1.0.1 Release Notes
- Kobold2D v1.0.0 Release Notes
- Kobold2D v0.9.6 Release Notes
- Kobold2D v0.9.5 Release Notes
- Kobold2D v0.9.4 Release Notes
- Kobold2D v0.9.3 Release Notes
- Kobold2D v0.9.2 Release Notes
- Kobold2D v0.9.1 Release Notes

Kobold2D v0.9.1 Release Notes

Important Changes

- Initial Release, everything is new.

Known Issues

- Incompatible with iOS SDK 5 **Preview**. See: Does Kobold2D work with the latest beta SDK?
- Project names with space(s) in them (eg "My Project") will report errors like

```
Prefix-iOS.pch:27:20: error: cocos2d.h: No such file or directory
```

- Not all template projects included yet (eg. Pinball game, Tilemap demos, UIKit Integration).
- Project Starter tool does not yet support adding projects to a custom workspace.
- Project Upgrade tool not yet included (not needed until preview 2).

Kobold2D v0.9.2 Release Notes

Important Changes and New Features

- Added Kobold2D Project Upgrader tool.
- Kobold2D Project Starter tool now allows you to select and create custom workspaces.
- Added two template projects: Orthogonal Tilemap & Pinball Game
- updated cocos3d to v0.6.1
- fixed failing builds due to spaces in project name or username

Bug Fixes and Minor Improvements

- added help button to tools
- Fixed: spaces in project names or path to Kobold2D (if moved from its default location) should no longer break the build with "missing file or directory" errors (pertaining to library headers)
- Fixed: Doodle Drop horizontal motion is inverted when playing in portrait upside down mode.
- Fixed: crash in FontLabel trying to load a .ttf font.
- Parallax side scroller doesn't appear correct on the iPad. Added note that background images weren't designed for iPad dimensions.
- Physics template projects not working correctly on Retina devices. Disabled Retina mode since these projects do not use HD images.

Known Issues

- Incompatible with iOS SDK 5 **Preview**. See: Does Kobold2D work with the latest beta SDK?
- Some template projects still missing (eg. UIKit demos, Iso Tilemap).
- Modified library code (eg. cocos2d-iphone) may not be linked into the App as it should be. This is an Xcode bug, see this thread for causes and solutions.

Kobold2D v0.9.3 Release Notes

The Most Important Changes & Additions

- Added Isometric Tilemap project template.
- Added Sprite Performance project template. Demonstrates the use and performance of CCSpriteBatchNode and PVR.CZZ texture atlas vs individual sprites from PNG files.
- Added Mac version of Doodle Drop.
- Added cocos2d-iphone-extensions develop branch (to support TMXGenerator)

Minor Improvements & Bug Fixes

- Fixed missing header search paths to cocos2d-iphone-extensions
- Fixed release build failure of Pinball game template
- Added missing "usesPremultipliedAlpha" call to Parallax Game template

Known Issues

- Incompatible with iOS SDK 5 **Preview**. See: Does Kobold2D work with the latest beta SDK?
- Modified library code (eg. cocos2d-iphone) may not be linked into the App as it should be. This is an Xcode bug, see this thread for causes and solutions.

Kobold2D v0.9.4 Release Notes

The Most Important Changes & Additions

- Added KKInput class to poll Mac keyboard & mouse states (Mac) and accelerometer, gyroscope and device motion (iOS)
- Added User Input template project to illustrate how to use KKInput
- Fixed archive builds not being able to create .IPA in Organizer.

- Fixed .c files causing compile errors due to mistake in prefix pch headers.
- updated iSimulate to v1.5 (new features)
- Compatibility with iOS 5 beta (projects build, but not fully tested)

Minor Improvements & Bug Fixes

- Fixed CocosDenshion CDXPropertyModifierAction not available in iOS builds
- Fixed Box2D headers in prefix headers now enclosed in `#ifdef __cplusplus`
- Fixed static library changes not being linked with app, unless build is "cleaned"
- Fixed Mac window: disable resize, remove "status bar" area at bottom, set AutoScale=NO as default, changed config.lua windowFrame to be the size of the GL view (takes title bar height into account)
- Mac info.plist: set KKApplicationMac as PrincipalClass to intercept Command key events
- LOG_EXPR(variable) now also works in Mac builds
- removed "strict selector matching" warning in template projects
- changed several template projects' first class to derive from CCLayer instead of CCScene

Known Issues

- KKInput touches and gesture recognition only partially implemented

Kobold2D v0.9.5 Release Notes

The Most Important Changes & Additions

- updated Box2D to 2.2.1 (from: 2.1.2)
  - Box2D API has changed! See Physics-Box2D template project for making the necessary changes to GLESDebugDraw, b2World initialization, screen border collision shape, etc in your Box2D projects.
- added "Cocos2D with UIKit Views" template project to demonstrate UIKit integration and hit testing (KKHitTest class)
- KKInput complete:
  - current touches exposed as CCArray with KKTouch objects. Locations are already converted to Cocos2D OpenGL view coordinates.
  - supports all standard gestures: tap, double-tap, long-press, swipe, pan, pinch, rotation.
- added AdMob banner support through config.lua settings (thanks to Simon Jewell)
  - AdMob support can be disabled in kkUserConfig.h to conserve approx. 900 KB (archive build app size)

Minor Improvements & Bug Fixes

- Build Settings: "Treat Warnings as Errors" defaults to NO to be easier on beginning developers and for prototyping / quick tests
  - Writing warning-free code is considered a (if not **the**) best practice.
- Added CCNode `+(id) nodeWithScene` method.
  - Returns autorelease instance of a node wrapped in a CCScene object, so that you don't have to write `+(id) scene` class method in layers anymore.
- Various fixes to Kobold2D API reference (ie added missing classes, split Kobold2D API reference into iOS & Mac OS)
- added KKAppDelegate methods to override root ViewController and the root UIView
  - removed RootViewController from all projects, it's no longer needed. You can now substitute your own subclass of KKRootViewController if necessary.
- removed the contents of the Frameworks groups (project cleanup)
- Kobold2D API reference now also exists in iOS and Mac OS versions
- use CocosDenshion SimpleAudioEngine in all tests instead of ObjectAL (CD is cross-platform)
- updated physics template projects to use KKInput, added gravity based on accelerometer values
- Kobold2D Project Upgrader tool now works on OS X 10.6 (Snow Leopard) and 32-bit machines
- Enabled Mac fullscreen support and added a config.lua setting `EnableFullScreen`
- OS X Deployment target now defaults to OS X 10.6 (was: 10.5)
- Added `EnableStatusBar` config.lua setting to show/hide the iPhone statusbar

Known Issues

Kobold2D v0.9.6 Release Notes

The Most Important Changes & Additions

- Kobold2D supports ARC!
  - ARC is enabled by default in all Kobold2D template projects.
  - Kobold2D and its template projects remain backward compatible to Xcode 4.1 or earlier, where ARC is unavailable.
  - Enabling ARC in already existing Kobold2D projects is simple.
- Library projects hidden by default (no more "builds but won't run" reports)

Minor Improvements & Bug Fixes

- Chipmunk SpaceManager Mac 64-Bit template enabled (fixed "freeze" bug)
- all frameworks moved from project build phase to common FrameworksAndLibraries.xcconfig
- acceleration values rawX, rawY, rawZ also available as x, y, z
- fix crash with iPhone Statusbar on iOS 3.0 to 3.1.3
- fix crash on iOS 3 due to CoreMotion.framework not weak linked
- fix crash with Box2D in release builds
- fix static analyzer reports in Kobold2D library code
- fix KKInput isAnyTouchOnNode:touchPhase: ignored the touchPhase
- fix panGestureTranslation was offset and inverted in portrait modes

Libraries updated

- Chipmunk SpaceManager (v0.1.2)
- Chipmunk (v6.0.2)
- Cocos3D (v0.6.2)

Known Issues

Kobold2D v1.0.0 Release Notes

The Most Important Changes & Additions

- Maintenance release, no new features but lots of bugfixes and tweaks.

Minor Improvements & Bug Fixes

- fix gesture pan, rotation and pinch velocity property not reset to 0.0 when fingers don't move
- fix wax headers in User Input template prefix headers not in `#ifdef ObjC` section
- fix crash connecting outlets in Cocos2D With UIKit template: MyView ViewController released too early
- fix KKGameKitHelper class being unavailable (linker error)
- fix KKInput anyTouchBegan/EndedThisFrame possibly being true over multiple frames if used with certain gestures, for example Double-Tap
- add header search path to FontLabel library
- fix mouse scroll wheel error on Mac OS X 10.6 due to calling hasPreciseScrollWheelDelta function available only in Mac OS X 10.7
- fix iAd crashes on devices with iOS 4.0 and 4.1
- fix iAd orientation issues when starting app in landscape mode
- added C functions: macOSVersionMajor(), macOSVersionMinor(), macOSVersionBugFix() to obtain the Mac OS X system version number at runtime through Gestalt.
- fixed a potential and sometimes intermittent linker error that some users reported and is apparently caused by missing i386 architecture for Simulator builds
- fixed "performSelector may cause a leak" warning in Pinball game project with ARC enabled

Libraries updated

- cocos2d-iphone-extensions v0.2
- cocos3d 0.6.3

Known Issues

- **Updating existing projects:** you may have to update your project's BuildSettings*.xcconfig files. To do that, create a new project with Kobold2D v1.0. Then replace the BuildSettings*.xcconfig files in your project with those from the newly created project to update your build settings.
- If you get: `file not found libarclite_macosx.a` then you are building a 64-Bit App under Snow Leopard with ARC enabled (which is the default). There may also be other build errors on Snow Leopard with Xcode 4.2 and building 64-Bit Mac Apps that are related to ARC. You need to disable Automatic Reference Counting (ARC) in the Build Settings of that project, because you can only build ARC apps for Mac OS X under Lion.

Kobold2D v1.0.1 Release Notes

The Most Important Changes & Additions

Kobold2D is now available on github. Note that several convenience features are missing from the github version. It is recommended to use the installer version if you're not an experienced git and Kobold2D user.

- KKAdBanner implements autorotation of Ad Providers (iAd & AdMob) thanks to Tomohisa. If an ad fails to load it will try to load an ad from the other enabled Ad Provider, thus increasing the chance of an Ad showing up at all. This is particularly important for countries where either iAd or AdMob won't show ads, or if one of the providers is (temporarily) unavailable.
- Added KKPixelMaskSprite for pixel-perfect collision detection.
- Added KKScreenshot for taking screenshots with CCRenderTexture (iOS only).

Minor Improvements & Bug Fixes

- FIX: using CCMenuItem* with blocks crashed when starting app
- FIX: Retina display might not have been initialized correctly even if enabled in config.lua
- FIX: KKInput resets its internal state when changing scenes. This fixes several issues, most notably touches that remained active after scene change even though the finger was lifted.
- FIX: If framerate is low touchBeganThisFrame events did not register on occassion. This happened if it was a fast tap or touch changed to moved state immediately after it began.
- FIX: gesturePanVelocity reported incorrect values and was never reset to 0,0 if touch wasn't moving.
- FIX: AdBanner views could disappear if UIViewController handled autorotation.
- FIX: frequent iAd warnings about "obstructed ads"
- IMP: Chipmunk & Box2 libaries have the "Relax IEEE Compliance" (GCC_FAST_MATH or -ffast-math) build setting enabled to speed up computations.
- IMP: KKGameKitHelper autorotates Game Center views to the current device orientation
- IMP: most KKGameKitHelper @protocol methods are now @optional
- IMP: added JRSwizzle class for internal method swizzling needs
- IMP: added CCDirector `frameCount` property which keeps track of the number of frames drawn since the app was started.
- IMP: added CCNode convenience method `intersectsNode:(CCNode*)other`
- IMP: added `anyTouchLocation` convenience property to KKInput
- IMP: added John Wordsworth's Box2DDebugLayer classto Box2D template project, renders Box2D debug info over existing nodes while still rendering the node's textures.

Libraries updated

- cocos3d 0.6.4

Known Issues

If you experience "ld file not found" and/or "failed with exit code 1" you should read Why do I get "ld .. failed with exit code 1"? for possible solutions.

If you get: `file not found libarclite_macosx.a` then you are building a 64-Bit App under Snow Leopard with ARC enabled (which is the default). There may also be other build errors on Snow Leopard with Xcode 4.2 and building 64-Bit Mac Apps that are related to ARC. You need to disable Automatic Reference Counting (ARC) in the Build Settings of that project, because you can only build ARC apps for Mac OS X under Lion.

KKAdBanner: if you enable ads you should also enable `AutorotationType = Autorotation.UIViewController` in config.lua so that ads are rotated correctly.

Kobold2D v1.0.2 Release Notes

The Most Important Changes & Additions

Maintenance release, only minor changes & additions.

Minor Improvements & Bug Fixes

- Relax IEEE Compliance (GCC_FAST_MATH) enabled for all targets to improve performance
- KKScreenshot clears the texture before rendering
- added KKScreenshot method `screenshotWithStartNode:(CCNode*)startNode` without saving to file, in case you only need the screenshot texture in memory (faster)
- CCNode extension method `transferToNode:(CCNode*)target` safely removes the node from its current parent and assigns it to the target node
- CCSprite extension method `spriteFromRenderTexture:(CCRenderTexture*)rtx`
- LOG_EXPR ARC compatibility fixes
- LOG_EXPR can now be used in both (Objective) C and (Objective) C++ code
- KKInput gives access to the underlying UIGestureRecognizer objects to be able to modify the behavior of gesture recognizers
- KKInput allows removal of individual touches: `[input removeTouch:touch]`. It's legal to remove a touch while iterating over touches.

- fix KKInput panGestureRecognizer translation and velocity values (depending on device orientation they had either x/y or +/- swapped)
- Separated Box2D & Chipmunk from cocos2d-iphone to be able to update them more easily and completely (cocos2d-iphone does not distribute all Chipmunk & Box2D files)
- Removed Wax library: Kobold2D used only a small portion of the Wax code to initialize Lua. The Lua library and initialization code have been merged into the Kobold2D library.
- ObjectAL: enabled the blocks API, updated documentation to include blocks methods.

Libraries updated

- Chipmunk 6.0.3
- cocos3d 0.6.5
- ObjectAL 2.1
- Wax removed

Known Issues

If you experience "ld file not found" and/or "failed with exit code 1" you should read Why do I get "ld .. failed with exit code 1"? for possible solutions.

If you get: `file not found libarclite_macosx.a` then you are building a 64-Bit App under Snow Leopard with ARC enabled (which is the default). There may also be other build errors on Snow Leopard with Xcode 4.2 and building 64-Bit Mac Apps that are related to ARC. You need to disable Automatic Reference Counting (ARC) in the Build Settings of that project, because you can only build ARC apps for Mac OS X under Lion.

KKAdBanner: if you enable ads you should also enable `AutorotationType = Autorotation.UIViewController` in config.lua so that ads are rotated correctly.

Kobold2D v1.0.4 Release Notes

The Most Important Changes & Additions

- Xcode 4.3 compatibility
- **SOLVED:** "file not found .../libkobold2d-ios.a" issue is caused if Xcode Build Location is set to the non-recommended setting `Locations Specified by Targets` (Xcode 4.3: `Legacy`). This problematic Xcode setting is now detected by Kobold2D during build and will alert the user to the issue to provide a quick fix and the recommended solution.

Minor Improvements & Bug Fixes

- KKInput: fix touches never being in "moved" phase
- Kobold2D Project Upgrader: fixed one issue that may have caused upgradeable projects not to be found
- Kobold2D Project Upgrader: logs .txt file where it looks for Kobold2D projects and how it decides to include them or not (will help debugging in case the above issue persists)
- Installer: tryfix of permission issue after installing (user has no write permission in Kobold2D folder). If still happening try running `sudo pkgutil --repair com.kobold2d` (for 1.0.2) respectively `sudo pkgutil --repair com.kobold2d.1.0.4` depending on version.
- Installer: prevent installer from removing previous version's files by changing the package id to contain the version number (ie com.kobold2d.1.0.4).
- iSimulate: disabled in template projects by default. Edit **BuildSettings-iOS.xcconfig** to enable iSimulate by uncommenting the **OTHER_LDFLAGS** line referencing (force-loading) iSimulate. This is to avoid the "incoming network connection" warning dialog caused by iSimulate which may be confusing or even deterring new users.
- Cocos3D: fixed a compiler error in CC3PODResources in Xcode 4.3 (missing cast)
- replaced donationware kobold.ttf font with completely free Ubuntu font

Libraries updated

Known Issues

KKAdBanner: if you enable ads you should also enable `AutorotationType = Autorotation.UIViewController` in config.lua so that ads are rotated correctly.

Kobold2D v1.0.5 Release Notes

The Most Important Changes & Additions

- same as v1.0.4 but includes a hotfix for build failures if the path to Kobold2D (ie your username) contains a space character.

Minor Improvements & Bug Fixes

Libraries updated

Known Issues

> KKAdBanner: if you enable ads you should also enable `AutorotationType = Autorotation.UIViewController` in config.lua so that ads are rotated correctly.

Kobold2D v1.1.0 Release Notes

The Most Important Changes & Additions

- uses cocos2d-iphone v1.1
- iPad 3 Retina support (-ipadhd file extension) thanks to cocos2d-iphone v1.1

Minor Improvements & Bug Fixes

- cocos3d and Hello Cocos3D template project removed due to incompatibility with cocos2d-iphone v1.1 & v2.0

Libraries updated

- cocos2d-iphone v1.1.0-beta2
- **REMOVED** cocos3d

Known Issues

> KKAdBanner: if you enable ads you should also enable `AutorotationType = Autorotation.UIViewController` in config.lua so that ads are rotated correctly.

Kobold2D v1.1.1 Release Notes

The Most Important Changes & Additions

- uses cocos2d-iphone version 1.1 with -ipadhd file extension support (version from May 3rd 2012)
- fixed KKInput missing events (specifically "this frame" variants) when scheduling selector with schedule:@selector(..) instead of scheduleUpdate
- 20%-40% faster compilation thanks to Prefix headers

Minor Improvements & Bug Fixes

- fixed installer permission issues (affected some user)
- updated project settings with recommended ARMv6 THUMB setting

Libraries updated

- cocos2d-iphone 1.1 (beta2b) (using current head revision, May 3rd 2012)
- Chipmunk 6.1.1
- Chipmunk SpaceManager 0.1.3

Known Issues

- Validate Settings in Xcode will mention that schemes ought to use LLDB instead of GDB. All project Run settings already set to use GBD and allowing the validation to make the changes automatically will not change the setting. To fix, edit scheme and change to LLDB manually.

KKAdBanner: if you enable ads you should also enable `AutorotationType = Autorotation.UIViewController` in config.lua so that ads are rotated correctly.

Kobold2D v1.1.2 Release Notes

The Most Important Changes & Additions

- compatible with Xcode 4.4

Minor Improvements & Bug Fixes

Libraries updated

Known Issues

Kobold2D v1.1.3 Release Notes

The Most Important Changes & Additions

- compatible with Xcode 4.5
- fixed: iOS 6 initial rotation issue
- iPhone 5 widescreen enabled in all template projects (see below how to enable for your app)
- Mountain Lion: no need to lower security level to run Installer. Package is now signed by "identified developer".
- NEW: Kobold2D Class Templates available, installed to Xcode. Use as basis for new CCNode class files. Includes stubs of frequently used methods: init, onEnter, cleanup, dealloc, update.

Minor Improvements & Bug Fixes

- added C function `isWidescreenEnabled()` to report if app is running on a widescreen device with widescreen "enabled" (ie when Default-568h@2x.png is included in the project)
- added CCNode category methods:
  - `removeChildrenInArray:(id<NSFastEnumeration>)childArray cleanup:(BOOL)cleanup` (removes the children of the given array)
  - `setPositionRelativeToParentPosition:(CGPoint)pos` (positions child node relative to parent's position, instead of lower-left corner of parent's texture - call with CGPointZero to center node on parent)

Libraries updated

Known Issues

- Existing projects must add a **Default-568h@2x.png** image with size of 640x1136 pixels to their project to enable iPhone 5 widescreen support.

Kobold2D v1.x.x Release Notes

This Kobold2D version is not available yet!
This page describes changes to Kobold2D which have been implemented, but the Kobold2D version it refers to has not been released yet.

The Most Important Changes & Additions

Minor Improvements & Bug Fixes

Libraries updated

Known Issues

Kobold2D v2.0.0 Release Notes

The Most Important Changes & Additions

- Kobold2D and all projects use cocos2d-iphone 2.0
- 20-40% faster compilation of libraries (cocos2d, chipmunk, etc) thanks to precompiled header
- Permission issues after installation should no longer occur

Minor Improvements & Bug Fixes

- Use Xcode target settings to choose the Supported Device Orientations.
- Autorotation is always enabled if there are at least two supported device orientations.

Libraries updated

- cocos2d-iphone 2.0 (beta 2)

Known Issues

- KKInput may miss touch began/ended events under some circumstances. Current workaround is to use regular touch events. Proper fix requires a rewrite of KKInput touch input handling.
- Cocos3D, Chipmunk SpaceManager are currently incompatible with cocos2d 2.0
    - Chipmunk SpaceManager Template project will not rotate the box sprites due to an incompatibility in SpaceManager.
- Deprecated config.lua properties: DirectorType, DirectorTypeFallback, AutorotationType, DeviceOrientation, AllowAutorotationOn1stAnd2ndGenerationDevices

General Info for all Cocos2D Users

Known Incompatibilities when upgrading to Cocos2D 2.0

- All gl commands need to be reviewed. Even though they compile, and may even seem to work, OpenGL will log errors. Replace with GLES 2.0 pendants, for example change `glColor4f` to `ccDrawColor`.
- The CCDirector deviceOrientation API has been removed. Use [UIDevice currentDevice].orientation property instead.
- All coordinates are in points, the "InPixels" properties are no longer available
- There's only one CCDirector type: CCDirectorDisplayLink. Setting Director type method has been removed.
- … and a lot more ...

When upgrading a Cocos2D 1.x project to Cocos2D 2.0

This lists only the things you need to do to get the basics working. It does not list all the API changes that can cause build errors.

- Add the fps-images-hd.png and fps-images-ipadhd.png files, otherwise FPS counter will be distorted on Retina displays.
- Update Info.plist, change required device capabilities from opengles-1 to opengles-2 and add armv7 to ensure iTunes disallows download of your app to incompatible devices, which would cause your app to be rejected during approval.
- You should remove all armv6 architecture references from Build Settings. Cocos2D 2.0 does not work with armv6 devices (1st & 2nd generation). Still building armv6 code will unnecessarily increase app size and may cause build errors in release builds.

Kobold2D v2.0.1 Release Notes

The Most Important Changes & Additions

- fix KKInput missing events (specifically "this frame" variants) if scheduling selectors with schedule:@selector(bla:) instead of scheduleUpdate.

Minor Improvements & Bug Fixes

Libraries updated

- Chipmunk 6.1.1
- Chipmunk SpaceManager 0.1.3

Known Issues

- Chipmunk SpaceManager is not fully compatible with cocos2d 2.0 (sprites won't rotate)
- Validate Settings in Xcode will mention that schemes ought to use LLDB instead of GDB. All project Run settings already set to use GBD and allowing the validation to make the changes automatically will not change the setting. To fix, edit scheme and change to LLDB manually.

Kobold2D v2.0.2 Release Notes

The Most Important Changes & Additions

- update to cocos2d-iphone 2.0 (final/stable)

Minor Improvements & Bug Fixes

Libraries updated

- cocos2d-iphone v2.0.0

Known Issues

- Chipmunk SpaceManager is not fully compatible with cocos2d 2.0 (sprites won't rotate)

Kobold2D v2.0.3 Release Notes

The Most Important Changes & Additions

- compatible with Xcode 4.4

Minor Improvements & Bug Fixes

Libraries updated

Known Issues

- Chipmunk SpaceManager is not fully compatible with cocos2d 2.0 (sprites won't rotate)

Kobold2D v2.0.4 Release Notes

The Most Important Changes & Additions

- compatible with Xcode 4.5
- fixed: iOS 6 initial rotation issue
- iPhone 5 widescreen enabled in all template projects (see below how to enable for your app)
- Mountain Lion: no need to lower security level to run Installer. Package is now signed by "identified developer".
- NEW: Kobold2D Class Templates available, installed to Xcode. Use as basis for new CCNode class files. Includes stubs of frequently used methods: init, onEnter, cleanup, dealloc, update.

Minor Improvements & Bug Fixes

- added C function `isWidescreenEnabled()` to report if app is running on a widescreen device with widescreen "enabled" (ie when Default-568h@2x.png is included in the project)
- added CCNode category methods:
  - `removeChildrenInArray:(id<NSFastEnumeration>)childArray cleanup:(BOOL)cleanup` (removes the children of the given array)
  - `setPositionRelativeToParentPosition:(CGPoint)pos` (positions child node relative to parent's position, instead of lower-left corner of parent's texture - call with CGPointZero to center node on parent)

Libraries updated

Known Issues

- Existing projects must add a **Default-568h@2x.png** image with size of 640x1136 pixels to their project to enable iPhone 5 widescreen support.
- Chipmunk SpaceManager is not fully compatible with cocos2d 2.0 (sprites won't rotate)

Kobold2D v2.1.0 Release Notes

The Most Important Changes & Additions

- compatible with Xcode 4.6 (fixed FIX_CATEGORY_BUG issue and others)
- improved App delegate to properly support iOS 6 autorotation

Minor Improvements & Bug Fixes

- enabled armv7s architecture (iPhone 5, iPod touch 5, iPad 3/4) in all targets

Libraries updated

- updated: cocos2d-iphone 2.1 rc0a
- updated: cocos2d-iphone-extensions 0.21 (latest from github)
- updated: Chipmunk 6.1.2
- updated: Chipmunk Spacemanager 0.2.01
- fixed: SneakyInput compiles with 2.1
- fixed: Chipmunk Spacemanager compiles with 2.1 (setOpacity & setColor without function)
- removed: AdMob SDK

Known Issues

- Chipmunk SpaceManager is not fully compatible with cocos2d 2.1 (color and opacity changes may not be applied to sprites etc handled by SpaceManager)

Kobold2D v2.x.x Release Notes

This Kobold2D version is not available yet!
This page describes changes to Kobold2D which have been implemented, but the Kobold2D version it refers to has not been released yet.

The Most Important Changes & Additions

Minor Improvements & Bug Fixes

Libraries updated

Known Issues

- Chipmunk SpaceManager is not fully compatible with cocos2d 2.0 (sprites won't rotate)